JOURNAL OF COMPUTER AND SYSTEM SCIENCES 15, 35–98 (1977)

Control Sets on Context-Free Grammar Forms*

S. A. GREIBACH

Department of System Science, University of California Los Angeles, Los Angeles, California 90024

Received March 15, 1976; revised November 19, 1976

For a context-free grammar form G, the result of the G-control operator acting on a family of languages $\mathscr{L}$ is defined as the family of languages formed by using members of $\mathscr{L}$ to control left-to-right derivations of all grammars which are interpretations of G. If G is left derivation bounded, the G-control operator takes a full semiAFL $\mathscr{L}$ into a full semiAFL which can be characterized by homomorphic replications on members of $\mathscr{L}$, and takes context-free full semiAFLs into quasi-realtime full semiAFLs, quasi-realtime full semiAFLs into quasi-realtime full semiAFLs and context-sensitive full semiAFLs into context-sensitive full semiAFLs. If G is nonterminal bounded and self-embedding and $\mathscr{L}$ is a full semiAFL not closed under the G-control operator then repeated applications of the G-control operator produce a strictly increasing chain of full semiAFLs. If G is not left derivation bounded the G-control operator can take the family of linear context-free languages onto the family of r.e. languages even if G is not self-embedding and all interpretations of G generate regular sets.

1. INTRODUCTION

Control sets on grammars were introduced by Ginsburg and Spanier [11] as one way of restricting the application of rules rather than the format of rules in a grammar and so investigating further the notion of a grammar. In this initial paper only left-to-right derivations were considered. The notion of a control set was extended to general derivations, and allied types of control mechanisms were studied by Salomaa [24, 25, 26], Rosenkrantz [23], and others. Recently the notion of controls on grammars was revived by E. Wotschke as a means of explaining the structure of natural languages without resorting to transformations [29].

Complementary to the notion of a control set as a method of modifying the generative power of a context-free grammar (and thus a class of control sets as a way of extending the class of context-free grammars), one can view a context-free grammar in its turn as a device for altering a language. In that view, each production in a grammar G is labeled by a distinct terminal symbol of a language C and G is regarded as a device for translating C into the language $L(G, C)$ of all words in $L(G)$ generated by left-to-right derivations of G labeled by words of C. Such a translation can be implemented by a pushdown store transducer and indeed to each grammar G with labeled productions there corresponds

* This paper was supported in part by the National Science Foundation under Grant No. DCR 74-15091.

ISSN 0022-0000

a deterministic one state pushdown store transducer T with $L(G, C) = T(C)$ for all languages C. Conversely each deterministic one state e-input free pushdown store transduction T corresponds to a (labeled) context-free grammar G effecting the same translation, while an arbitrary pushdown store transduction T corresponds to a grammar G and a finite state transducer M with $T(C) = L(G, M(C))$ everywhere. (This can be seen directly or by combining results on control sets in [11] with results in [31] on multitape transducers.) Regarded in this light it is perhaps a misnomer to call C a "control set" on G since G is rather a grammar-directed translator operating on C; however, we use the "control set" terminology in this paper since it is already accepted in the literature.

Extending this viewpoint from grammars and languages to classes of grammars and classes of languages, we can regard a class of grammars $\mathscr{G}$ as an operator, the $\mathscr{G}$-control operator, on a family of languages $\mathscr{L}$, taking $\mathscr{L}$ into a family we can define as

$$\text{CONTROL}(\mathscr{G}, \mathscr{L}) = \{L(G, C) \mid G \text{ in } \mathscr{G}, C \text{ in } \mathscr{L}\}.$$

Thus one can study the effect of a class of grammatical translators on a class of languages. This idea already appears in the literature. For example, Ginsburg and Spanier observed [11] that the regular control operator (i.e., regular grammars) takes any full AFL into itself, that the context-free control operator (i.e., the whole class of context-free grammars) takes any AFL $\mathscr{L}$ into $\{h(L_1 \cap L_2) \mid L_1$ context-free, $L_2 \in \mathscr{L}$, h homomorphism$\}$ and that the restricted context-free control operator (i.e., context-free grammars with a uniform bound on the number of left-to-right steps before a new terminal appears leftmost) takes any full AFL $\mathscr{L}$ into

$$\{h(L_1 \cap L_2) \mid L_1 \text{ context-free}, L_2 \in \mathscr{L}, h \text{ nonerasing homomorphism}\}.$$

By contrast, Khabbaz showed using specialized techniques that the linear control operator (i.e., linear context-free grammars) takes the family of context-free languages into a proper subset of the family of context-sensitive languages and repeated iteration of that operator starting with the regular sets produces an infinite hierarchy of families of languages [21, 22]; we shall see that these results can be dramatically extended.

As a simpler example, consider the class of grammars with productions of the form $S \to Sa$, labeled a, and $S \to e$ labeled $\mathscr{S}$. If a is allowed to range over a vocabulary Σ not including $\mathscr{S}$ and $C \subseteq \Sigma^*$, then such a grammar G takes $C\mathscr{S}$ into $L(G, C)$ which is C reversed. So this class of grammars can be thought of as a class of reversal operators. Indeed, if $\mathscr{G}$ is the class of left linear grammars (productions of the form $X \to Yu$, $X \to u$, u a terminal string, X, Y nonterminals) it is not hard to see that for any full semiAFL $\mathscr{L}$, CONTROL$(\mathscr{G}, \mathscr{L})$ is the class of reversals of members of $\mathscr{L}$ and so $\mathscr{G}$ can be considered to be the reversal operator (on full semiAFLs). Notice that to obtain such results one needs both a class of grammars, so that, for example, one is not restricted to a particular alphabet, and a class of languages, so that one is not too much tied to special grammatical devices like ad hoc rules such as $S \to e$.

To put this concept in a more definite framework we need to decide what restrictions to place on "class of languages" and on "class of grammars." For "class of languages" the

concept of a full semiAFL (nontrivial family of languages closed under union and finite state transductions) seems reasonable, not only because full semiAFLs have attractive mathematical properties and have been previously studied in such abstract settings but also because union and finite state transductions are operations occurring naturally in this study and are vital tools.

It is harder to define an "abstract family of grammars" or "class of similar grammars" and, in the opinion of this writer, no altogether satisfactory definition exists to date. Possibly no such definition can exist because at least two of the types of results one would like to obtain, on one hand nice algebraic properties of the languages generated and on the other hand reasonable relationships with parsing and syntax directed translations, are probably incompatible.

However, to study the effects of the type of control operators we have been discussing, a few requirements on a class of grammars $\mathscr{G}$ seem natural. We want to be free of dependence on particular alphabets, both terminal and nonterminal, and on particular labeling of productions. On the other hand, we wish to be able to keep distinct various well-known families: right linear, left linear, linear, nonterminal bounded, derivation bounded, and so forth. Thus if G is in $\mathscr{G}$, $\mathscr{G}$ should contain all variants of G formed by just changing the identity of the terminals or substituting a terminal string for a single terminal symbol but not ones formed by inserting terminals in new positions (or else right linear, left linear, and linear grammar families would be identified). Similarly, $\mathscr{G}$ should be closed under renaming nonterminals but not under inserting new nonterminals (or else linear grammars would be identified with general grammars) or changing distinct nonterminals into the same nonterminal (or else nonterminal bounded grammars could become grammars not even derivation bounded).

The concept of a "grammar form" and a "grammatical family" was introduced by Gabrielian and Ginsburg [6] and Cremers and Ginsburg [5] as one way of unifying some of the underlying properties of well-known families of grammars and trying to impart to the study of grammars and the languages they generate the structure that AFL theory has brought to the study of machines and the languages they accept. To each grammar G regarded as a grammar form a family $\mathscr{G}(G)$ of grammars interpreting G is associated in Ref. [5]; a formal definition of $\mathscr{G}(G)$ appears in Section 2 of this paper. It turns out that $\mathscr{G}(G)$ has just the "closure" properties informally outlined above; a formal definition of $\mathscr{G}(G)$ appears in Section 2 of this paper. Hence we regard a "class of grammars" as either a class $\mathscr{G}(G)$ generated by one grammar or, more generally (as in the study of nonterminal bounded grammars), as a class $\mathscr{G}$ of grammars such that whenever G is in $\mathscr{G}$, $\mathscr{G}(G) \subseteq \mathscr{G}$.

For a grammar G, the result of the G-control operator acting on a family of languages $\mathscr{L}$ is defined (in Section 2) as CONTROL$(\mathscr{G}(G), \mathscr{L})$. The class of languages generated by a family $\mathscr{G}(G)$ of grammars is denoted $\mathscr{L}(G)$. We consider grammars G_1 and G_2 to be equivalent as grammar forms (g-equivalent) if $\mathscr{L}(G_1) = \mathscr{L}(G_2)$. We consider G_1 and G_2 to be control set equivalent (c-equivalent), i.e., equivalent as grammatical transducers, if CONTROL$(\mathscr{G}(G_1), \mathscr{L})$ = CONTROL$(\mathscr{G}(G_2), \mathscr{L})$ for every full semiAFL $\mathscr{L}$.

To see some simple examples, notice that the regular grammars are $\mathscr{G}(G_0)$ for $G_0 = (\{S, a\}, \{a\}, \{S \to aS, S \to a\}, S)$, the left linear grammars are $\mathscr{G}(G_R)$ for $G_R =$

$(\{S, a\}, \{a\}, \{S \to Sa, S \to a\}, S)$ while the linear context-free grammars are $\mathscr{G}(G_L)$ for $G_L = (\{S, a\}, \{a\}, \{S \to aSa, S \to a\}, S)$. Now $\mathscr{L}(G_0) = \mathscr{L}(G_R)$ since both right and left linear grammars generate all regular languages, so G_0 and G_R are obviously g-equivalent. But for any full semiAFL $\mathscr{L}$, CONTROL$(\mathscr{G}(G_0), \mathscr{L}) = \mathscr{L}$ while CONTROL$(\mathscr{G}(G_R), \mathscr{L})$ is the family of reversals of members of $\mathscr{L}$ (as mentioned above) and there are full semiAFLs not closed under reversal. So G_0 and G_R are not c-equivalent! Thus c-equivalence is a much finer distinction than g-equivalence.

Something even stranger occurs when we look at the grammar $G_1 = (\{S, X, B, a\}, \{a\}, \{S \to aS, S \to X, X \to XB, X \to e, B \to e\}, S)$. The nonterminal B generates the empty word and nothing else. If we consider only the languages generated by members of $\mathscr{G}(G)$ (g-equivalence) we may as well drop all productions except $S \to aS$ and add $S \to a$; it is not hard to see that $\mathscr{L}(G)$ is also the family of regular languages. But if we consider the effect of applying the G-control operator on the family LINEARL of linear context-free languages, something quite surprising emerges. For CONTROL$(\mathscr{G}(G_1)$, LINEARL) is equal to RE, the family of recursively enumerable languages! The proof is roughly similar to the proof that every recursively enumerable language is the homomorphic image of the intersection of two linear context-free languages [1]. Thus this grammar G_1 is equivalent to a regular grammar as a language generating device but as a control set operator has power more akin to the power of the general context-free control operator.

The strange pathology of the grammar just mentioned comes in part from the rules of the form $X \to XB$. Since B generates only the empty word, no new terminal strings are generated. However, in left-to-right derivations rules of this type can pile up arbitrarily many nonterminals which cannot be controlled until much later. A left derivation bounded grammar is one which has a uniform bound on the number of nonterminals which can appear in any word in a left-to-right derivation. Grammars like G are not left derivation bounded. This proves to be the sharp cutpoint between grammar families exhibiting the power of the general context-free control operator (as described in [11]) and those exhibiting the more limited power of the linear context-free control operator (as described in [21, 22] for some special cases). Namely, we show (Theorem 4.4) that either a grammar G is c-equivalent to a left derivation bounded grammar or CONTROL$(\mathscr{G}(G)$, LINEARL) = RE and the two cases are mutually exclusive. The left derivation bounded condition turns out to be very significant in other connections, as seen in Sections 2, 3, and 7. This is not surprising since we are considering left-to-right derivations; if we used right-to-left derivations we would expect right derivation bounded grammars to play a key role while for general derivations nonterminal bounded grammars would have the central place.

The two main themes of this study which constantly interweave are characterizations of CONTROL$(\mathscr{G}(G), \mathscr{L})$ (in terms of $\mathscr{L}(G)$, $\mathscr{L}$, and various operations) for particular restrictions on grammar G and conditions for c-equivalence of G to grammars in particular formats. Thus the result cited above (Theorem 4.4) gives both a characterization for CONTROL$(\mathscr{G}(G), \mathscr{L})$ when G is not left derivation bounded and LINEARL $\subseteq \mathscr{L} \subseteq$ RE (namely, CONTROL$(\mathscr{G}(G), \mathscr{L})$ = RE) and also gives a necessary and sufficient condition under which G is never c-equivalent to a left derivation bounded grammar, namely, CONTROL$(\mathscr{G}(G)$, LINEARL) = RE.

A related necessary and sufficient condition for G to be or not to be c-equivalent to a left derivation bounded grammar appears in Section 2. We see that if G is left derivation bounded and C belongs to a full semiAFL $\mathscr{L}$, then $L(G, C)$ can be expressed as a linearly bounded homomorphic image of the intersection of a context-free language and a member of $\mathscr{L}$ (Theorems 2.9 and 2.10). This tells us that if G is left derivation bounded and $\mathscr{L}$ is a full semiAFL contained in CS, the family of context-sensitive languages, then CONTROL($\mathscr{G}(G)$, $\mathscr{L}$) $\subseteq$ CS (Theorem 2.11). But together with Theorem 4.4 it says that G is c-equivalent to a left derivation bounded grammar if and only if CONTROL($\mathscr{G}(G)$, LINEARL) $\subseteq$ CS. This gives us a sharp division in complexity properties for grammar operators; either CONTROL($\mathscr{G}(G)$, LINEARL) = RE or else CONTROL ($\mathscr{G}(G)$, LINEARL) is not only properly contained in CS but in fact in the family of checking automaton languages (cf. [33]).

Types of grammars which provide normal forms for context-free grammars may or may not be normal forms under c-equivalence. On one hand, sequential grammars are not canonical for context-free languages (there are context-free languages not generable by any sequential context-free grammar) but every context-free grammar is c-equivalent to a sequential context-free grammar (the catch is that G's being sequential is not inherited by members of $\mathscr{G}(G)$) (Theorem 2.14). On the other hand, Chomsky Normal Form and Greibach Normal Form are canonical for context-free grammars and for g-equivalence (every context-free grammar is g-equivalent to one in Chomsky Normal Form and to one in Greibach Normal Form) but not for c-equivalence. In Section 5 we see examples of grammars not c-equivalent to grammars in weak versions of Chomsky Normal Form or Greibach Normal Form. For example, the grammar G_R described above is not c-equivalent to any grammar in weak Chomsky Normal Form or in weak Greibach Normal Form or even to any left recursion free grammar. The grammar $G_2 = (\{S, B, a\}, \{a\}, \{S \to aSB,\ B \to e,\ S \to a\}, S)$ is left recursion free but not c-equivalent to any grammar in weak Greibach Normal Form, while the grammar G_L above is in weak Greibach Normal Form, (namely, terminals leftmost in productions) but is not c-equivalent to any grammar in Greibach Normal Form. Expression in these normal forms is intimately connected with the question of when CONTROL($\mathscr{G}(G)$, $\mathscr{L}$) has the characterization

(**) For every full semiAFL $\mathscr{L}$,

$$\text{CONTROL}(\mathscr{G}(G), \mathscr{L}) = \{h(L_1 \cap L_2) \mid h \text{ homomorphism}, L_1 \in \mathscr{L}(G), L_2 \in \mathscr{L}\}.$$

The major results of Section 5 are (1) (**) is primarily a characterization for non left derivation bounded grammars and holds for a left derivation bounded grammar G if and only if G is c-equivalent to a regular grammar (Theorem 5.1); (2) (**) holds for G if and only if G is c-equivalent to a grammar in Greibach Normal Form (Theorem 5.14); and (3) (**) holds for G if $\mathscr{L}(G)$ is the whole family of context-free languages (Theorem 5.16).

For left derivation bounded grammars we have characterizations of CONTROL ($\mathscr{G}(G)$, $\mathscr{L}$) in terms of $\mathscr{L}$ and types of homomorphic replications. For example, for LINEARG, the family of linear context-free grammars we have CONTROL (LINEARG, $\mathscr{L}$) = $\{\{h_1(w)\, h_2(w^R) \mid w \in L\} \mid L \in \mathscr{L},\ h_1, h_2$ homomorphisms$\}$ for any

semiAFL $\mathscr{L}$ (Theorem 6.1). More generally, if G is nonterminal bounded and self-embedding and $\mathscr{L}$ is a semiAFL, then the closure of $\mathscr{L}$ under the G-control operator is the closure of $\mathscr{L}$ under homomorphic replication (Theorem 7.1). This not only provides a characterization of CONTROL($\mathscr{G}(G)$, $\mathscr{L}$) for G nonterminal bounded, but it also gives a necessary and sufficient condition for G to be c-equivalent to a nonterminal bounded grammar. Namely, G is c-equivalent to a nonterminal bounded grammar if and only if for every full semiAFL $\mathscr{L}$, CONTROL($\mathscr{G}(G)$, $\mathscr{L}$) is contained in the closure of $\mathscr{L}$ under homomorphic replication and G is c-equivalent to a self-embedding nonterminal bounded grammar if and only if for every full semiAFL $\mathscr{L}$, the closure of $\mathscr{L}$ under the G-control operator is equal to its closure under homomorphic replication (Theorem 7.3). A similar situation obtains for left derivation bounded grammars and an infinite homomorphic replication operator, yielding both a characterization of the closure of $\mathscr{L}$ under the G-control operator (Theorem 7.10) and a necessary and sufficient condition for a grammar to be c-equivalent to a left derivation bounded grammar (Corollary 7.12).

The paper is organized as follows. Section 2 reviews the notation for grammars and a-transducers, and gives formal definitions for semiAFLs, grammar forms, control sets, and c-equivalence. Then we establish a series of technical propositions concerning which manipulations on grammars preserve c-equivalence; the results are used frequently in the subsequent chapters. As an important consequence we obtain the results cited above (Theorem 2.9 and 2.11) that for a left derivation bounded grammar G the G-control operator takes context-sensitive full semiAFLs into context-sensitive full semiAFLs (and in Section 4 we see that Theorem 2.9 is the best possible result of that nature).

In Section 3 we examine closure properties of CONTROL($\mathscr{G}(G)$, $\mathscr{L}$) and notice in particular (Theorem 3.5) that if G is left derivation bounded and nontrivial (i.e., generates an infinite language) and $\mathscr{L}$ is a semiAFL, then CONTROL($\mathscr{G}(G)$, $\mathscr{L}$) is a full semiAFL. This also holds for weaker assumptions on G but it is not known whether it is true for every nontrivial context-free grammar G.

Section 4 considers the pathology of grammars which are not left derivation bounded and centers on the result, mentioned above, that CONTROL($\mathscr{G}(G)$, LINEARL) = RE if and only if G is not c-equivalent to any left derivation bounded grammar.

Section 5 focuses on when the characterization (**) described above holds.

In Sections 6 and 7 we turn to the characterization of CONTROL($\mathscr{G}(G)$, $\mathscr{L}$) when G is left derivation bounded. Section 6 concentrates on the family of linear context-free grammars, LINEARG. First we characterize CONTROL(LINEARG, $\mathscr{L}$) (the linear control operator acting on $\mathscr{L}$) for a semiAFL $\mathscr{L}$ in terms of homomorphic replications of members of $\mathscr{L}$ (Theorem 6.1). Then we notice that CONTROL(LINEARG, $\mathscr{L}$) as a consequence has certain special syntatic properties (Lemmas 6.2–6.4). One very strong corollary is that the linear control operator preserves noninclusion among full semiAFLs (Theorem 6.5). If $\mathscr{L}$ is a full semiAFL not equal to CONTROL(LINEARG, $\mathscr{L}$), then every iteration of application of the linear control operator increases the family of languages obtained (Corollary 6.6), and CONTROL(LINEARG, $\mathscr{L}$) is never closed under concatenation (Theorem 6.11). Hence we obtain a very strong hierarchy theorem (Theorem 6.13) which is independent of the particular family of control sets under discussion.

We continue in Section 7 by extending the results to general nonterminal bounded

grammars. Instead of establishing precise characterization results for each type of nonterminal bounded grammar, we establish a general connection between controls on linear grammars and controls on nonterminal bounded grammars. In particular if G is nonterminal bounded, then for any semiAFL $\mathscr{L}$, CONTROL($\mathscr{G}(G)$, $\mathscr{L}$) is contained in the closure of $\mathscr{L}$ under the linear control operator (Theorem 7.1). We show that this result characterizes nonterminal bounded grammars (Theorem 7.3). Then we use Theorem 7.1 to establish general hierarchy results akin to those for linear grammars (Theorems 7.4, 7.5, 7.6, and 7.8). Finally we conclude by giving a general characterization result for controls on arbitrary left derivation bounded grammars in terms of a type of infinite homomorphic replication operator (Theorem 7.10), followed by a weak hierarchy theorem (Theorem 7.17).

Section 8 summarizes our results and mentions some open questions.

2. Definitions and Basic Properties

In this section we concentrate on establishing a series of technical results which allow us to manipulate context-free grammars while preserving c-equivalence. First we must give the necessary formal definitions.

We start by quickly reviewing the definitions of context-free grammars, of derivations, and of control sets in order to establish a common notation. We assume the reader to be familiar with basic notions and facts concerning formal languages as found, e.g. in Salomaa [24].

Notation. We represent a *context-free grammar* as a quadruple $G = (V, \Sigma, P, S)$ where V is a finite vocabulary, Σ is a subset of V, S is in $V - \Sigma$, and P is a finite set of *rules* or *productions* of the form $X \to u$ for X in $V - \Sigma$, and u in V^*.[1] Members of Σ are called *terminals*, members of $V - \Sigma$ are called *nonterminals*, and S is called the *start* symbol. A rule $X \to u$ is an *X-rule*.

We shall assume that the rules of P are all labeled, each with a distinct name. Often it will be convenient to assume that the rules name themselves. If p: $X \to u$ is a rule in P and x and y are in V^* we write $xXy \Rightarrow_G xuy$. If x is in Σ^* (is a terminal string) we also write $xXy \overset{p}{\Rightarrow}_G xuy$. We define $\overset{*}{\Rightarrow}_G$, $\overset{+}{\Rightarrow}_G$, and $\overset{z}{\Rightarrow}_G$ inductively from $\overset{*}{\Rightarrow}$ and $\overset{p}{\Rightarrow}_G$. For any string w in V^* we write $w \overset{*}{\Rightarrow}_G w$ and $w \overset{e}{\Rightarrow}_G w$. If $w_1 \overset{*}{\Rightarrow}_G w_2$ and $w_2 \Rightarrow_G w_3$, then $w_1 \overset{*}{\Rightarrow}_G w_3$ and $w_1 \overset{+}{\Rightarrow}_G w_3$. If $w_1 \overset{u}{\Rightarrow}_G w_2$ and $w_2 \overset{v}{\Rightarrow}_G w_3$, then $w_1 \overset{uv}{\Rightarrow}_G w_3$. We write $w_1 \overset{L^*}{\Rightarrow} w_2$ if $w_1 \overset{v}{\Rightarrow}_G w_2$ for any v; if $v \neq e$ we also write $w_1 \overset{L_+}{\Rightarrow}_G w_2$. Whenever the grammar G is obvious from the context, then we omit the subscript G.

If $w_1 \overset{*}{\Rightarrow} w_2$, then w_2 is *derivable* from w_1 or w_1 *generates* w_2. If $w_1 \overset{z}{\Rightarrow} w_2$, we call this a *left-to-right* or *leftmost derivation* from w_1 to w_2 with *control word* z. If w_2 is a terminal string, the derivation is *completed*.

Definition. The *language generated by* $G = (V, \Sigma, P, S)$ is the set

$$L(G) = \{w \in \Sigma^* \mid S \overset{*}{\Rightarrow} w\}.$$

[1] For a set A, A^* is the free monoid generated by A with identity e (the empty string).

The *language generated by G with control set C* is the set

$$L(G, C) = \{w \in \Sigma^* \mid \exists z \text{ in } C,\ S \overset{z}{\Rightarrow} w\}.$$

DEFINITION. For any family $\mathscr{G}$ of grammars, let

$$\mathscr{L}(\mathscr{G}) = \{L(G) \mid G \in \mathscr{G}\}.$$

DEFINITION. For any family $\mathscr{G}$ of grammars and any family $\mathscr{L}$ of control sets, let

$$\text{CONTROL}(\mathscr{G}, \mathscr{L}) = \{L(G, C) \mid G \in \mathscr{G}, C \in \mathscr{L}\}.$$

We shall be primarily concerned with context-free grammars so the term "grammar" will be synonymous with "context-free grammar" unless otherwise modified. We want to designate certain special types of grammars.

DEFINITION. A grammar $G = (V, \Sigma, P, S)$ is *reduced* if for each A in V there are words u and v such that $S \overset{*}{\Rightarrow} uAv$ and for each X in $V - \Sigma$ there is a word w in Σ^* such that $X \overset{*}{\Rightarrow} w$.

DEFINITION. A grammar $G = (V, \Sigma, P, S)$ is *linear* if all of its rules are of the forms $X \to \alpha Y \beta$ and $X \to \gamma$ for α, β, and γ in Σ^* and Y in $V - \Sigma$. If G is linear then $L(G)$ is a *linear context-free language*. If all of the rules of G are of the forms $X \to \alpha Y$ and $X \to \gamma$ for α and γ in Σ^* and Y in $V - \Sigma$, then G is *regular* and $L(G)$ is a *regular language*.

We need notation for these commonly used families of grammars and languages.

DEFINITION. Let CFG be the family of context-free grammars and CFL the family of context-free languages. Let REGULARG be the family of regular grammars and REGULARL the family of regular languages. Let LINEARG be the family of linear grammars and LINEARL the family of linear context-free languages.

Two operations that will be important to us are those of substitution and of a-transducer mapping.

DEFINITION. Let Σ be a finite vocabulary and for each a in let $\tau(a)$ be a language. Let $\tau(e) = \{e\}$ and for u, v in Σ^*, let $\tau(uv) = \tau(u)\tau(v)$. For $L \subseteq \Sigma^*$, let $\tau(L) = \bigcup_{w \in L} \tau(w)$. Then τ is a *substitution*. If $\tau(a)$ is regular for each a in Σ then τ is *regular*; if $\tau(a)$ is finite for each a in Σ then τ is *finite* and if $\tau(a)$ is e-free (does not contain e) for each a in Σ, then τ is *e-free*.

DEFINITION. An *a-transducer* is a 6-tuple $M = (K, \Sigma, \Delta, H, q_0, F)$ such that K is a finite set of *states*, Σ is a finite *input* vocabulary, Δ is a finite *output* vocabulary, q_0 in K is the *initial* or *start* state, and $F \subseteq K$ is the subset of *final* or *accepting* states, and H is a finite subset of $K \times \Sigma^* \times \Delta^* \times K$; members of H are called *transitions*. We define a relation $\vdash_M$ (or $\vdash$ where M is understood) on $K \times \Sigma^* \times \Delta^*$ by $(q, uv, z) \vdash_M (p, v, zx)$

if (q, u, x, p) is in H, and let $\vdash_M$ be the transitive reflexive closure of $\vdash_M$. The *output of M for input w* is the set $M(w) = \{z \mid \exists p \in F, (q_0, w, e) \vdash_M (p, e, z)\}$. For a language L, $M(L) = \{z \mid \exists w \in L, p \in F, (q_0, w, e) \vdash_M^* (p, e, z)\}$ and $M^{-1}(L) = \{w \mid M(w) \cap L \neq \phi\}$. We call M acting on L an *a-transducer mapping* (of L) and M^{-1} an *inverse a-transducer* maping. If $H \subseteq K \times \Sigma^* \times \Delta^+ \times K$ then M is *e-output free*. If $H \subseteq K \times \Sigma^+ \times \Delta^* \times K$ then M is *e-input free*.

The a-transducer mapping is perhaps the most general form of a finite state tranduction. We shall assume the reader to be familiar with the power of a-transducers and shall generally describe them by giving their actions rather than by detailing the transitions.

DEFINITION. A *semiAFL* is a family of languages containing at least one nonempty language and closed under nonerasing homomorphism,[2] inverse homomorphism, intersection with regular sets, and union. A *full semiAFL* is a semiAFL closed under homomorphism. An *AFL* is a semiAFL closed under concatenation and Kleene $+$.[3] A *full AFL* is an AFL closed under homomorphism.

DEFINITION. For a family of languages $\mathscr{L}$ let $\mathscr{M}(\mathscr{L})$ (respectively $\hat{\mathscr{M}}(\mathscr{L})$, $\mathscr{F}(\mathscr{L})$, $\hat{\mathscr{F}}(\mathscr{L})$) be the least semiAFL (respectively full semiAFL, AFL, full AFL) containing $\mathscr{L}$. If $\mathscr{L} = \{L\}$ for an individual language L, write $\mathscr{M}(L)$ (respectively $\hat{\mathscr{M}}(L)$, $\mathscr{F}(L)$, $\hat{\mathscr{F}}(L)$) instead of $\mathscr{M}(\{L\})$ (respectively $\hat{\mathscr{M}}(\{L\})$, $\mathscr{F}(\{L\})$, $\hat{\mathscr{F}}(\{L\})$) and call it the *principal* semiAFL (respectively full semiAFL, AFL, full AFL) generated by L.

We shall frequently use the facts (cf. [8]) that for any nonempty language L,

$$\mathscr{M}(L) = \{M(L) \mid L \in \mathscr{L}, M \text{ is an } e\text{-output} - \text{free } a\text{-transducer}\},$$

and

$$\hat{\mathscr{M}}(L) = \{M(L) \mid L \in \mathscr{L}, M \text{ is an } a\text{-transducer}\}.$$

Now we are ready for our definitions of grammar forms. First we define complete interpretations of grammars.

DEFINITION. A *complete interpretation* of a grammar $G = (V, \Sigma, P, S)$ is a pair (G', τ) such that $G' = (V', \Sigma', P', S')$ is a grammar and τ is a finite substitution on V^* such that $\tau(V - \Sigma) \subseteq V' - \Sigma'$, $\tau(\Sigma) \subseteq (\Sigma')^*$, $S' \in \tau(S)$, $\tau(X) \cap \tau(Y) = \phi$ for X, Y in $V - \Sigma$ and $X \neq Y$ and $P' = \{Z' \to u \mid \exists Z \to u \in P, Z' \in \tau(Z), u' \in \tau(u)\}$. For a rule $Z \to u$ in P we write $\tau(Z \to u) = \{Z' \to u' \mid Z' \in \tau(Z), u' \in \tau(u)\}$; we write $P' = \tau(P)$. We also write $G' = \tau(G)$ and call G' a *complete interpretation* of G.

DEFINITION. A grammar (V', Σ', P', S) is a *subgrammar* of (V, Σ, P, S) if $\Sigma' \subseteq \Sigma$, $V' - \Sigma' \subseteq V - \Sigma$, and $P' \subseteq P$.

DEFINITION. A grammar G' is an *interpretation* of a grammar G if it is a subgrammar of some complete interpretation of G.

[2] A homomorphism h is *nonerasing* if $h(w) \neq e$ whenever $w \neq e$.

[3] Kleene $+$ is the operation taking L into $L^+ = LL^*$.

DEFINITION. For a grammar of G, the *grammatical family generated by* G is the family $\mathscr{G}(G) = \{H \mid H \text{ is an interpretation of } G\}$. For a grammar G let

$$\mathscr{L}(G) = \mathscr{L}(\mathscr{G}(G)) = \{L(H) \mid H \in \mathscr{G}(G)\}.$$

For a family of grammars $\mathscr{G}$ we can consider CONTROL($\mathscr{G}$, $\mathscr{L}$) to be the result of the *$\mathscr{G}$-control operator* acting on the family of languages $\mathscr{L}$. Thus if $\mathscr{L}$ = CONTROL($\mathscr{G}$, $\mathscr{L}$) we say that $\mathscr{L}$ is *closed under the $\mathscr{G}$-control operator*. If $\mathscr{G} = \mathscr{G}(G)$ for a grammar G, we speak instead of the *G-control operator*.

There is an unpleasant ambiguity possible in the definition of CONTROL($\mathscr{G}$, $\mathscr{L}$). What do we use for the names of productions in members of $\mathscr{G}$ and how do we know that these names match up with the symbols used in the languages of $\mathscr{L}$? One way out is to assume that all our families of languages are closed under alphabetical variations (i.e., under one-one length preserving homomorphisms); we shall adopt this convention.

If G' is a subgrammar of G and the vocabulary of C does not contain as symbols names of productions in G but not in G', obviously $L(G, C) = L(G', C)$. Thus as far as the G-control operator is concerned we can limit attention to complete interpretations of G. If $\mathscr{L}$ is closed under alphabetical variations then we see that CONTROL($\mathscr{G}(G)$, $\mathscr{L}$) = $\{L(H, C) \mid C \in \mathscr{L}, H \text{ is a complete interpretation of } G\}$.

The following useful proposition is immediate from the definitions and from the fact that an interpretation of an interpretation is an interpretation [5].

PROPOSITION 2.1. *For any family of languages $\mathscr{L}$, if G' is a member of $\mathscr{G}(G)$, then*

$$\text{CONTROL}(\mathscr{G}(G'), \mathscr{L}) \subseteq \text{CONTROL}(\mathscr{G}(G), \mathscr{L}).$$

Using the standard cross-product construction employed to show the family of context-free languages closed under intersection with regular sets one has the next proposition.

PROPOSITION 2.2. *For any grammar G,* CONTROL($\mathscr{G}(G)$, REGULARL) = $\mathscr{L}(G)$.

DEFINITION. Grammars G and G' are *equivalent as grammar forms, g-equivalent*, written $G \equiv^g G'$ if $\mathscr{L}(G) = \mathscr{L}(G')$. They are *equivalent as control grammar forms, c-equivalent*, written $G \equiv^c G'$ if CONTROL($\mathscr{G}(G)$, $\mathscr{L}$) = CONTROL($\mathscr{G}(G')$, $\mathscr{L}$) for every full semiAFL $\mathscr{L}$. Write $G \leqslant^c G'$ if CONTROL($\mathscr{G}(G)$, $\mathscr{L}$) $\subseteq$ CONTROL($\mathscr{G}(G')$, $\mathscr{L}$) for every full semiAFL $\mathscr{L}$.

Clearly g-equivalence and c-equivalence are equivalence relations and $\leqslant^c$ is transitive. Proposition 2.2 shows that c-equivalence implies g-equivalence; we shall see that the converse is not necessarily true.

If G' is the reduced subgrammar of G then G is c-equivalent to G' since derivations which are not completable cannot affect $L(G, C)$. Thus we can assume without loss of generality that we are dealing only with reduced grammars as our grammar forms.

There are various results in [5] which allow one to manipulate grammars within the same g-equivalence class. We need some similar results for c-equivalence. Unfortunately,

the analogs of results for g-equivalence do not always hold for c-equivalence and when they do, the proofs are more tedious. First we need a result allowing us to add a production $X \to w$ whenever w can be derived from X.

PROPOSITION 2.3. *Let G be a grammar, let $\mathscr{L}$ be a full* semiAFL, *let $X \Rightarrow_G^+ w$ and let G' be the grammar formed from G by adding production $X \to w$. Then*

(a) $\text{CONTROL}(\{G'\}, \mathscr{L}) \subseteq \text{CONTROL}(\mathscr{G}(G), \mathscr{L})$,

and

(b) $\text{CONTROL}(\mathscr{G}(G'), \mathscr{L}) = \text{CONTROL}(\mathscr{G}(G), \mathscr{L})$, *and $G \equiv^c G'$, a fortiori.*

Proof. First observe that to establish (b) it suffices to do so in the special case where w is derivable from X in two steps, since we can then obtain the general case using repeated applications of Proposition 2.1 and the special case.

Now consider $L(G', C)$ for C in $\mathscr{L}$ and productions $X \to uYv$ and $Y \to z$ in G with $w = uzv$. For convenience we can assume that productions name themselves. If u contains only terminals we form C' by substituting $(X \to uYv)(Y \to z)$ for $(X \to uzv)$ everywhere in C and notice that $L(G', C) = L(G, C')$ and C' is in $\mathscr{L}$.

Otherwise we form G'' in $\mathscr{G}(G)$ by adding a new nonterminal Y' and new productions $X \to uY'v$ and $Y' \to z$. Let τ be the e-free regular substitution $\tau(X \to uzv) = (Y' \to z)^*(X \to uY'v)(Y' \to z)^*$ and $\tau(p) = (Y' \to z)^* p (Y' \to z)^*$ for any production $p \neq (X \to uzv)$. Then $L(G', C) = L(G'', \tau(C))$ which is in $\text{CONTROL}(\mathscr{G}(G), \mathscr{L})$.

Next consider any complete interpretation $\tau(G')$ of G'. Clearly $\tau(G')$ is the grammar formed from $\tau(G)$ by adding the rule set $\{X' \to w' \mid X' \in \tau(X), w' \in \tau(w)\}$. Also, if $X' \in \tau(X)$ and $w' \in \tau(w)$, then $X' \Rightarrow_{\tau(G)}^+ w'$. Thus by repeated use of the proof of (a) and Proposition 2.1 we obtain $\text{CONTROL}(\{\tau(G')\}, \mathscr{L}) \subseteq \text{CONTROL}(\mathscr{G}(\tau(G)), \mathscr{L}) \subseteq \text{CONTROL}(\mathscr{G}(G), \mathscr{L})$ and thus $\text{CONTROL}(\mathscr{G}(G'), \mathscr{L}) \subseteq \text{CONTROL}(\mathscr{G}(G), \mathscr{L})$. By Proposition 2.1, $\text{CONTROL}(\mathscr{G}(G), \mathscr{L}) \subseteq \text{CONTROL}(\mathscr{G}(G'), \mathscr{L})$ and thus we have established (b). ∎

Notice that Proposition 2.3 could be strengthened by replacing "let $\mathscr{L}$ be a full semiAFL" by "let $\mathscr{L}$ be a family of languages closed under e-free regular substitution."

Now we wish to be able to eliminate a production $X \to u$ in favor of directly producing from X whatever could be produced from u. We have to do this with care, as the operation does not always preserve c-equivalence. First let us define what it means for a grammar to have the replacement property. In what follows a derivation $X \Rightarrow^+ y$ "passes through" a word w if the derivation can be written

$$X \Rightarrow y_1 \Rightarrow y_2 \Rightarrow \cdots \Rightarrow y_m = y$$

for $m \geqslant 1$ and for some i, $1 \leqslant i \leqslant m$, $w = y_i$.

DEFINITION. A grammar G has the *replacement property with respect to a nonterminal X and a finite set of words W* if G is c-equivalent to the grammar formed by replacing the X-rules of G with the rule set $\{X \to w \mid w \in W\}$. It has the *replacement property* if it has the replacement property with respect to every nonterminal X and every finite set of

words W such that every completed left-to-right derivation from X passes through a word of W and every member of W can be generated from X.

Not every grammar has the replacement property. We shall give two conditions on grammars which imply the replacement property. First we give a sufficient condition on X and W for G to have the replacement property with respect to X and W.

PROPOSITION 2.4. *Let X be a nonterminal in a grammar G, let $n \geqslant 1$ be an integer and W a finite set of words such that every member of W is derivable from X and every completed left-to-right derivation from X has an initial subderivation of the form*

$$X \overset{L}{\Rightarrow} w_1 \overset{L}{\Rightarrow} \cdots \overset{L}{\Rightarrow} w_m = w$$

with $m \geqslant 1$, w in W and $| w_i | \leqslant n$ for $1 \leqslant i \leqslant m$.[4] *Let G' be formed from G by replacing the X-rule by $\{X \to w \mid w \in W\}$. Then G has the replacement property with respect to X and W; i.e., $G \equiv^c G'$.*

Proof. We start by establishing the following property (*) for any G, X, W, and G' satisfying the hypotheses above.

(*) For every full semiAFL $\mathscr{L}$, CONTROL($\{G\}$, $\mathscr{L}$) $\subseteq$ CONTROL($\{G'\}$, $\mathscr{L}$).

Let $\mathscr{L}$ be a full semiAFL. Consider $L(G, C)$ for C in $\mathscr{L}$. We shall construct an a-transducer M such that $L(G, C) = L(G', M(C))$ so $L(G, C)$ is in CONTROL($\{G'\}$, $\mathscr{L}$). The idea is that machine M is to replace control words for subderivations from X by appropriate productions $X \to w$ and otherwise act as the identity. Now in any completed left-to-right derivation from X a word w in W must be encountered before any word of length $n + 1$ or greater has been derived from X. Hence M need only record words shorter in length than $n + 1$ in its finite state control. So M can have initial and final state q_0 plus all states of the form $\langle y \rangle$ for y a string over the vocabulary of G of length less than or equal to n. The transition set is

$$\begin{aligned}
&\{(q_0, p, p, q_0) \mid p \text{ is a production of } G \text{ which is not an } X\text{-rule}\} \\
&\quad \cup \{(q_0, X \to u, e, \langle u \rangle) \mid X \to u \text{ is an } X\text{-rule}, 0 \leqslant | u | \leqslant n\} \\
&\quad \cup \{(\langle u \rangle, p, e, \langle v \rangle) \mid p \text{ is a production of } G, u \overset{p}{\Rightarrow} v, 0 \leqslant | u |, | v | \leqslant n\} \\
&\quad \cup \{(\langle w \rangle, e, X \to w, q_0) \mid w \in W\}.
\end{aligned}$$

Thus we have established (*).

Now consider an arbitrary complete interpretation (H, τ) of G. Every left-to-right derivation in H corresponds under τ to one in G [5]. Thus if $X' \in \tau(X)$, $W' = \{w \in \tau(W) \mid X' \overset{*}{\Rightarrow} w\}$, and $k = \text{Max}(\{1\} \cup \{| y | \mid \exists \text{ terminal symbol } a, y \in \tau(a)\})$, then H, W', X' and kn satisfy the hypotheses of this proposition. So if $\tau(X) = \{X_1, \ldots, X_m\}$,

[4] For a word w, $| w |$ is the length of w.

$W_i = \{w \in \tau(W) \mid X_i \overset{*}{\Rightarrow}_H w\}$, $H_0 = H$ and for $1 \leqslant i \leqslant m$, H_{i+1} is H_i with the X_i-rules replaced by $\{X_i \to w \mid w \in W_i\}$, repeated applications of (*) yield

$$\text{CONTROL}(\{H_i\}, \mathscr{L}) \subseteq \text{CONTROL}(\{H_{i+1}\}, \mathscr{L}).$$

Hence

$$\text{CONTROL}(\{H\}, \mathscr{L}) \subseteq \text{CONTROL}(\{H_m\}, \mathscr{L}),$$

but $H_m = \tau(G')$ is in $\mathscr{G}(G')$. So $\text{CONTROL}(\mathscr{G}(G), \mathscr{L}) \subseteq \text{CONTROL}(\mathscr{G}(G'), \mathscr{L})$. By Proposition 2.3, we may assume that G already contains $\{X \to w \mid w \in W\}$. Hence applying Proposition 2.1 completes the proof. ▮

There are two major conditions under which the hypotheses of Proposition 2.4 always hold for suitable X and W. One is the absence of erasing rules, i.e., rules of the form $X \to e$.

COROLLARY 2.5. *If G is a grammar without erasing rules, then G has the replacement property.*

Proof. If G has no erasing rules then for any X and w, if $X \Rightarrow^{L*} u \Rightarrow^{L*} w$, then $|u| \leqslant |w|$. So for suitable nonterminal X and word set W one applies Proposition 2.4 with $n = \text{Max}(\{1\} \cup \{|w| \mid w \in W\}.)$ ▮

Another simple condition is the existence of a general bound on the number of nonterminals in left-to-right derivations, the left derivation bounded condition, introduced by Walljasper [28].

DEFINITION. A nonterminal X in a grammar G is *left derivation bounded* (*ldb*) if there is an integer $k \geqslant 0$ such that every word left-to-right derivable in G from X contains at most k nonterminals; then k is a *left derivation bound for X in G*. Grammar G is *left derivation bounded* (*ldb*) if every nonterminal in G is ldb; k is a *left derivation bound for G* if it is a left derivation bound for every nonterminal in G.

DEFINITION. Let LDBG be the family of ldb grammars.

If G has left derivation bound k and $X \Rightarrow^{L*} u \Rightarrow^{L*} w$, then certainly $|u| \leqslant k + |w|$ since terminals are not erased and u contains at most k nonterminals. Thus we have the following corollary.

COROLLARY 2.6. *If G is* ldb *it has the replacement property.*

Proposition 2.4 allows us to dispense with standstill rules, i.e., rules of the form $X \to Y$ for X and Y nonterminals.

COROLLARY 2.7. *Any grammar is c-equivalent to one without standstill rules.*

Proof. If G is a grammar with a nonterminal X in $V - \Sigma$ and standstill X-rules, then apply Proposition 2.4 to

$$W = \{X \to u \mid X \to u \text{ is an } X\text{-rule in } G \text{ and } u \text{ is not in } V - \Sigma\}$$
$$\cup \{X \to u \mid \exists Y \in V - \Sigma,\ Y \text{ can be derived from } X \text{ using only nonerasing rules, } Y \to u \text{ is a } Y\text{-rule with } u \text{ not in } V - \Sigma\}.$$

and $n = \text{Max}(\{1\} \cup \{|\ w\ | \mid w \in W\})$. ∎

The existence of erasing rules (rules $X \to e$) and of symbols which are not ldb may result in a grammar without the replacement property. Consider G_1 with rules $S \to aS$, $S \to X$, $X \to XB$, $B \to e$, and $X \to e$; G_2 with rules $S \to aS$, $S \to X$, and $X \to e$; and G_3 with rules $S \to aS$, $S \to a$, and $S \to e$. We can apply Proposition 2.4 to G_2, S, and $W_S = \{aS, a, e\}$ to see that $G_2 \equiv^c G_3$. Clearly $\mathscr{G}(G_3)$ is the family of regular grammars and so for any full semiAFL $\mathscr{L}$, CONTROL($\mathscr{G}(G_3)$, $\mathscr{L}$) = $\mathscr{L}$ [11]. But we shall see in Section 5 that CONTROL($\mathscr{G}(G_1)$, LINEARL) is the family of recursively enumerable languages and so not equal to CONTROL($\mathscr{G}(G_2)$, LINEARL) = CONTROL($\mathscr{G}(G_3)$, LINEARL) = LINEARL. So G_1 does not have the replacement property with respect to X and $W_X = \{e\}$ and thus does not have the replacement property. The problem is that although every completed derivation from X must reach the empty string eventually, along the way the intermediate strings can become arbitrary large and the set of control words leading from X to e is not a regular set. Also note that G_1 is in $\mathscr{G}(G_0)$ for the grammar G_0 with rules $S \to aS$, $S \to X$, $X \to XB$, $B \to a$, and $X \to a$ and G_0 has no erasing rules and thus does have the replacement property. So the replacement property is not inheritable; a grammar G can have it and some members of $\mathscr{G}(G)$ not have it. On the other hand, if G is ldb, so is every member of $\mathscr{G}(G)$ and thus in this case members of $\mathscr{G}(G)$ do inherit the replacement property from G.

For grammars with ldb symbols we can establish a stronger version of the replacement property.

PROPOSITION 2.8. *Let G be a grammar with a rule $X \to uYv$ such that Y is a nonterminal and every nonterminal in u is* ldb. *Let $\{Y \to y_1, \ldots, Y \to y_m\}$ be the set of Y-rules. Let G' be the grammar formed by replacing $X \to uYv$ in G by the rule set $\{X \to uy_iv \mid 1 \leqslant i \leqslant m\}$. Then G is c-equivalent to G'.*

Proof. First we establish by an argument very similar to the one in Proposition 2.4, the following statement.

(*) For G and G' satisfying the hypotheses of this proposition and any full semiAFL $\mathscr{L}$, CONTROL($\{G\}$, $\mathscr{L}$) $\subseteq$ CONTROL($\{G'\}$, $\mathscr{L}$).

Let $\mathscr{L}$ be a full semiAFL and consider $L(G, C)$ for C in $\mathscr{L}$. This time we wish to construct an a-transducer which upon encountering $X \to uYv$ nondeterministically replaces it with some rule $X \to uy_iv$ and later verifies that rule $Y \to y_i$ was supposed to be applied

to the occurrence of Y in uYv. Note that X does not appear in u or any descendant of u so $X \to uYv$ will not be used again until after u is processed. We can keep track of the nonterminals appearing after $X \to uYv$ and left of Y since the ldb property puts a bound on the number of such symbols in any step of a left-to-right derivation.

Let h be the homomorphism which erases terminals and is the identity on nonterminals. Let k be an ldb bound for the nonterminals in u and let $n = k \mid u \mid$. The state set of M consists of initial state q_0 which is the only accepting state plus all states $\langle v, i \rangle$ for $1 \leqslant i \leqslant m$ and v a string of nonterminals no longer than n. The transition set of M is

$$\{(q_0, p, p, q_0) \mid p \text{ is a rule of } G,\ p \neq (X \to uYv)\}$$

$$\cup \{(q_0, X \to uYv, X \to uy_iv, \langle h(u), i \rangle) \mid 1 \leqslant i \leqslant m\}$$

$$\cup \{(\langle v, i \rangle, p, p, \langle h(z), i \rangle) \mid p \text{ is a rule of } G,\ v \overset{p}{\Rightarrow} z,\ 0 \leqslant \mid v \mid,\ \mid h(z) \mid \leqslant n\}$$

$$\cup \{(\langle e, i \rangle, Y \to y_i, e, q_0) \mid 1 \leqslant i \leqslant m\}.$$

Then $L(G, C) = L(G', M(C))$, which establishes (*).

Now if (H, τ) is any complete interpretation of G, the nonterminals of all members of $\tau(u)$ must be ldb in H. Thus we can use (*) repeatedly as we did in Proposition 2.4 to show that for any full semiAFL $\mathscr{L}$,

$$\text{CONTROL}(\{H\}, \mathscr{L}) \subseteq \text{CONTROL}(\{\tau(G')\}, \mathscr{L}).$$

Hence $G \leqslant^c G'$. Using Proposition 2.3 we can assume that $\{X \to uy_iv \mid 1 \leqslant i \leqslant m\}$ is already in G and so by Proposition 2.1, $G' \leqslant^c G$. ∎

By applying Propositions 2.4 and 2.8 repeatedly we can eliminate erasing rules in an ldb grammar in a very strong fashion.

DEFINITION. A grammar $G = (V, \Sigma, P, S)$ is *monotonic* if S does not appear on the right-hand side of any rule and for every rule $Z \to u$ in P either $u \in \Sigma$ or $\mid u \mid \geqslant 2$ or $Z = S$ and $u = e$.

If G is monotonic, it tells us quite a bit about the complexity of $L(G, C)$ for "nice" control sets.

THEOREM 2.9. *If G is an* ldb *grammar, we can effectively find a monotonic* ldb *grammar G' such that*

$$\text{CONTROL}(\{G\}, \mathscr{L}) \subseteq \text{CONTROL}(\{G'\}, \mathscr{L})$$

for every full semiAFL $\mathscr{L}$, *and G is c-equivalent to G'.*

Proof. Let $G = (V, \Sigma, P, S)$ be an ldb grammar. First observe that by adding a new start symbol if necessary we can certainly assume that S never appears on the right-hand

side of any rule. For each nonterminal X in $V - \Sigma$ such that $X \overset{*}{\Rightarrow} e$ (i.e., X can be erased), let $\bar{X}$ be a new symbol and let I be the set of all such new symbols and let $\tau(X) = \{X, \bar{X}\}$. Elsewhere in V define substitution τ by $\tau(A) = \{A\}$. We form a grammar $\bar{G}$ from G by splitting every symbol X that can be erased in G into two representatives: the unbarred original X which now cannot be erased unless $X = S$ and a barred copy which must be erased. We define a substitution σ on P by setting for $X \neq S$

$$\sigma(X \to u) = \{X \to v \mid v \in \tau(u) - I^*\} \cup \{\bar{X} \to v \mid v \in \tau(u) \cap I^*\}$$

while

$$\sigma(S \to u) = \{S \to v \mid v \in \tau(u)\}.$$

Let $\bar{P} = \sigma(P)$. Let $\bar{G} = (V \cup I, \Sigma, \bar{P}, S)$. For any control, set C clearly $L(G, C) = L(\bar{G}, \sigma(C))$. Thus CONTROL($\{G\}$, $\mathscr{L}$) $\subseteq$ CONTROL($\{\bar{G}\}$, $\mathscr{L}$) for every full semiAFL $\mathscr{L}$ Clearly $\bar{G}$ is in $\mathscr{G}(G)$, and is ldb, and $G \equiv^c \bar{G}'$.

Next form H from $\bar{G}$ by replacing all $\bar{X}$-rules by $\bar{X} \to e$ for every $\bar{X}$ in I; grammar H is also ldb. Every completed left-to-right derivation from any barred symbol $\bar{X}$ must end in the empty string so since $\bar{G}$ is ldb we can apply statement (*) in the proof of Proposition 2.4 repeatedly to show that for every full semiAFL $\mathscr{L}$,

$$\text{CONTROL}(\{\bar{G}\}, \mathscr{L}) \subseteq \text{CONTROL}(\{H\}, \mathscr{L}).$$

Proposition 2.4 also shows that $\bar{G} \equiv^c H$.

Now we need Proposition 2.8. Let H' be formed from H by replacing every occurrence of a barred symbol on the right-hand side of any rule by the empty string, i.e., by erasing all barred symbols from the right-hand side of rules. By repeated applications of statement (*) in the proof of Proposition 2.8 we see that

$$\text{CONTROL}(\{H\}, \mathscr{L}) \subseteq \text{CONTROL}(\{H'\}, \mathscr{L})$$

for every semiAFL $\mathscr{L}$. Also, $H \equiv^c H'$. Finally, the barred symbols are now useless in H' so if we remove them we get a grammar H'' satisfying the definition of monotonicity except perhaps for standstill rules and obviously $L(H', C) = L(H'', C)$ everywhere, and $H' \equiv^c H''$.

Finally, we can repeatedly apply statement (*) in the proof of Proposition 2.4 to ldb grammar H'' as in Corollary 2.7 to eliminate standstill rules. We obtain an ldb monotonic grammar G' such that

CONTROL($\{G\}, \mathscr{L}$) $\subseteq$ CONTROL($\{G'\}, \mathscr{L}$) for every full semiAFL $\mathscr{L}$, and $G \equiv {}^cG'$. ∎

When we examine the complexity of $L(G, C)$ we see that Theorem 2.9 cannot be significantly improved. The grammar must be c-equivalent to an ldb grammar and $\mathscr{L}$ must be a full semiAFL. First let us apply to Theorem 2.9 a result in [11] regarding controls on monotonic grammars. To do so, we need some additional notation.

DEFINITION. For a family of languages $\mathscr{L}$ let

$$\mathscr{H}(\mathscr{L}) = \{h(L) \mid L \in \mathscr{L},\ h \text{ is a nonerasing homomorphism}\},$$
$$\hat{\mathscr{H}}(\mathscr{L}) = \{h(L) \mid L \in \mathscr{L},\ h \text{ is a homomorphism}\},$$
$$\mathscr{H}^{\text{lin}}(\mathscr{L}) = \{h(L) \mid L \in \mathscr{L},\ h \text{ is a homomorphism linear bounded on } L\}.^{5}$$

DEFINITION. For families of languages $\mathscr{L}_1$ and $\mathscr{L}_2$, let

$$\mathscr{L}_1 \wedge \mathscr{L}_2 = \{L_1 \cap L_2 \mid L_1 \in \mathscr{L}_1, L_2 \in \mathscr{L}_2\}.$$

DEFINITION. Let MONCFG be the family of monotonic context-free grammars. We quote the following theorem from Ref. [11].

THEOREM 2.10. *For any full* semiAFL $\mathscr{L}$,

$$\text{CONTROL(MONCFG, } \mathscr{L}) \subseteq \mathscr{H}^{\text{lin}}(\text{CFL} \wedge \mathscr{L}).$$

Thus Theorems 2.9 and 2.10 tell us that in particular

$$\text{CONTROL(LDBG, } \mathscr{L}) \subseteq \mathscr{H}^{\text{lin}}(\text{CFL} \wedge \mathscr{L}) \qquad \text{for any full semiAFL } \mathscr{L}.$$

For context-free control sets we know that $\mathscr{H}^{\text{lin}}(\text{CFL} \wedge \text{CFL})$ is contained in the family of *quasi-realtime languages*, i.e. those languages accepted in realtime by nondeterministic multitape Turing machines [3].

DEFINITION. Let QUASI be the family of quasi-realtime languages, CS the family of context-sensitive languages, and RE the family of recursively enumerable languages.

We have the following corollary which is important enough to state as a theorem. (Recall that CS is closed under intersection and linear erasing [30] as is QUASI [3].)

THEOREM 2.11. (a) *For any full* semiAFL $\mathscr{L}$,

$$\text{CONTROL(LDBG, } \mathscr{L}) \subseteq \mathscr{H}^{\text{lin}}(\text{CFL} \wedge \mathscr{L}).$$

(b) For any full semiAFL $\mathscr{L}$ such that $\mathscr{L} \subseteq \text{CS}$,

$$\text{CONTROL(LDBG, } \mathscr{L}) \subsetneq \text{CS}.$$

(c) For any full semiAFL $\mathscr{L}$ such that $\mathscr{L} \subseteq \text{QUASI}$

$$\text{CONTROL(LDBG, } \mathscr{L}) \subsetneq \text{QUASI}.$$

(d) For context-free control sets,

$$\text{CONTROL(LDBG, CFL)} \subsetneq \text{QUASI} \subseteq \text{CS}.$$

[5] A homomorphism h is *linear bounded* on a language L if there is a $k \geqslant 1$ such that $|w| \leqslant k \operatorname{Max}(1, |h(w)|)$ for all w in L.

Note that CONTROL(LDBG, $\mathscr{L}$) is obviously closed under homomorphism while neither QUASI or CS is so closed; hence the improper inclusions above. It is not known whether QUASI is properly contained in CS.

In Section 5 we shall see that if G is not c-equivalent to an ldb grammar then CONTROL($\mathscr{G}(G)$, LINEARL) = RE which of course properly contains QUASI. Thus we cannot remove the restriction that G be c-equivalent to an ldb grammar.

We also cannot remove the restriction that the semiAFL $\mathscr{L}$ be full unless G is trivial.

DEFINITION. A nonterminal X in a grammar G is *nontrivial* if infinitely many terminal strings can be generated from X in G. A grammar G is *nontrivial* if $L(G)$ is infinite.

If G is trivial every member of $\mathscr{G}(G)$ is also trivial [5] and so CONTROL($\mathscr{G}(G)$, $\mathscr{L}$) can only contain finite sets regardless of the identity of $\mathscr{L}$. Otherwise, CONTROL ($\mathscr{G}(G)$, $\mathscr{L}$) must contain at least $\hat{\mathscr{M}}(\mathscr{L})$ or $\hat{\mathscr{M}}(\mathscr{L}^R)$ if $\mathscr{L}$ is a semiAFL.[6]

LEMMA 2.12. *If G is a nontrivial grammar and $\mathscr{L}$ is a* semiAFL, *then either* CONTROL($\mathscr{G}(G)$, $\mathscr{L}$) *contains* $\hat{\mathscr{M}}(\mathscr{L})$ *or* CONTROL($\mathscr{G}(G)$, $\mathscr{L}$) *contains* $\hat{\mathscr{M}}(\mathscr{L}^R)$.

Proof. Since G is nontrivial, there must be subderivations in G of the forms $S \overset{*}{\Rightarrow} uXv$, $X \overset{*}{\Rightarrow} xXy$, and $X \overset{*}{\Rightarrow} z$ for S the start symbol of G, X some nonterminal (possibly S), u, v, x, y, and z terminal strings and xy nonempty. Using Proposition 2.3, we may assume that G contains $S \to uXv$, $X \to xXy$, and $X \to z$. There are two cases depending on whether x or y is nonempty. First suppose that x is nonempty. Any member of $\hat{\mathscr{M}}(\mathscr{L})$ can be expressed as $h(d_1Ld_2)$ for a language d_1Ld_2 in $\mathscr{L}$, a vocabulary Σ and special symbols c, d_1, and d_2 not in Σ and a homomorphism h which is the identity on Σ and erases c, d_1, and d_2. Then $\mathscr{G}(G)$ contains a grammar G' with nonterminal vocabulary $\{S, X\}$, terminal vocabulary Σ, rules $S \to X$ labeled d_1, $X \to e$, labeled d_2, $X \to X$ labeled c and for every a in Σ a rule $X \to aX$ labeled a. Then $h(d_1Ld_2)$ is equal to $L(G', d_1Ld_2)$ and so is in CONTROL($\mathscr{G}(G)$, $\mathscr{L}$). If objection is raised to the use of a production $X \to X$, a slightly more complex construction will do using $X \to Y$ and $Y \to X$ for a new symbol Y.

If y is nonempty the argument is similar. We notice that this time the rule in G' labeled a must be $X \to Xh(a)$ and so $L(G', d_1Ld_2) = (h(d_1Ld_2))^R$ and we can turn up an arbitrary member of $\hat{\mathscr{M}}(\mathscr{L}^R)$. ∎

Thus, for example, if G is a nontrivial grammar, CONTROL($\mathscr{G}(G)$, QUASI) contains the closure of QUASI under homomorphism which is RE. On the other hand, CONTROL(MONCFG, QUASI) = QUASI. Hence there must be left derivation bounded grammars G such that CONTROL($\{G\}$, QUASI) cannot be contained in CONTROL($\{G'\}$, QUASI) for any monotonic grammar G'. So we see that Theorem 2.9 can only apply to grammars c-equivalent to left derivation bounded grammars and to full semiAFLs.

We give one final proposition which is useful in establishing c-equivalence for left derivation bounded grammars. It says that if G is ldb then in determining which languages

[6] For a word $w = a_1 \cdots a_n$, each a_i a symbol, the *reversal* of w is $w^R = a_n \cdots a_1$; $e^R = e$. For a language L the *reversal* of L is $L^R = \{w^R \mid w \in L\}$ and for a family of languages $\mathscr{L}$, $\mathscr{L}^R = \{L^R \mid L \in \mathscr{L}\}$.

are in CONTROL($\mathscr{G}(G)$, $\mathscr{L}$) for a semiAFL $\mathscr{L}$, we do not have to consider all members of $\mathscr{G}(G)$ or even all complete interpretations of G. It suffices to consider only interpretations of G which leave nonterminal symbols unchanged.

DEFINITION. A complete interpretation σ of a grammar D is *simple* if $\sigma(Z) = \{Z\}$ for every nonterminal Z. Let $\mathscr{G}_s(G) = \{\sigma(G) \mid \sigma \text{ is a complete simple interpretation of } G\}$.

PROPOSITION 2.13. *Let G be left derivation bounded. Then for any* semiAFL $\mathscr{L}$,

$$\text{CONTROL}(\mathscr{G}(G), \mathscr{L}) = \text{CONTROL}(\mathscr{G}_s(G), \mathscr{L}).$$

Proof. Let $G = (V, \Sigma, P, S)$. We have already remarked that

$$\text{CONTROL}(\mathscr{G}(G), \mathscr{L}) = \text{CONTROL}(\mathscr{G}_c(G), \mathscr{L})$$

where $\mathscr{G}_c(G) = \{\tau(G) \mid \tau \text{ is a complete interpretation of } G\}$.

Let $H = \tau(G)$ be a complete interpretation of G. Let σ be the corresponding simple interpretation of G defined by $\sigma(a) = \tau(a)$ for every terminal a and $\sigma(Z) = \{Z\}$ for every nonterminal Z.

Let C be a control set in $\mathscr{L}$. We shall construct an a-transducer M such that $L(H, C) = L(\sigma(G), M(C))$. The idea is that by recording the sequence of nonterminals in its memory and using productions of $\sigma(G)$ to get the "right" terminal strings in the "right" places, M can simulate derivations of H. Since H is left derivation bounded, there is a uniform upper bound on the number of nonterminals that can appear in any word generated from left to right from the start symbol and so M can record the nonterminal sequence in its finite state control.

Initially M knows it is simulating a nonterminal sequence containing only the start symbol of H. Consider a time when M knows the nonterminal sequence of H is X, $U_0, \ldots, U_r$, $r \geqslant 0$, and M reads production p in its input. If p is not an X-rule, M blocks. Suppose p is the X-rule

$$X \to \alpha_0 Y_1 \cdots \alpha_{s-1} Y_s \alpha_s$$

where $s \geqslant 0$, the α_i are terminal strings and if $s = 0$ the rule is $X \to \alpha_0$. Now in G we must have a rule

$$X' \to \beta_0 Z_1 \cdots \beta_{s-1} Z_s \beta_s$$

such that $X \in \tau(X')$, each Y_i is in the corresponding $\tau(Z_i)$ and each α_i is in $\tau(\beta_i)$. Then $\sigma(G)$ has the rule

$$\hat{p}\colon X' \to \alpha_0 Z_1 \cdots \alpha_{s-1} Z_s \alpha_s .$$

So M prints $\hat{p}$ and records in its memory the new nonterminal sequence

$$Y_1, \ldots, Y_s, U_0, \ldots, U_r .$$

If the nonterminal sequence is empty when M reaches the end of its input then it accepts and gives output; if it empties prematurely, M blocks. Hence

$$L(H, C) = L(\sigma(G), M(C)) \in \text{CONTROL}(\mathscr{G}_s(G), \mathscr{L}). \quad \blacksquare$$

Remark. The left derivation bounded condition is essential in Proposition 2.13. Consider the grammar G_0 with production set $\{S \to SS, S \to a\}$. Now $\mathscr{G}(G_0)$ contains every context-free grammar in Chomsky Normal Form, so CONTROL($\mathscr{G}(G_0)$, REGULARL) = CFL [11]. But it is not difficult to see that CONTROL($\mathscr{G}_s(G_0)$, REGULARL) is the family, COUNT, of one counter language (cf. [16] for definitions). The family of one counter languages cannot be expressed as $\mathscr{L}(\mathscr{G})$ for any grammar G [5] and thus not as CONTROL($\mathscr{G}(G)$, REGULARL); in the notation of [5], COUNT is not a *grammatical family*. However, we have expressed COUNT as CONTROL($\mathscr{G}_S(G)$, REGULARL). Another example is the family of single-turn one counter languages (cf. [16] for definitions) which is CONTROL($\mathscr{G}_s(G_1)$, REGULARL) for the grammar G_1 with production set $\{S \to aSX, S \to a, X \to a\}$. Details are left to the interested reader.

We conclude this section by observing that sequential grammars are canonical for c-equivalence as well as for g-equivalence.

DEFINITION. A grammar $G = (V, \Sigma, P, S)$ is *sequential* if whenever $X \overset{*}{\Rightarrow} uYv$ and $Y \overset{*}{\Rightarrow} u'Xv'$, then $X = Y$.

THEOREM 2.14. *Every context-free grammar is c-equivalent to a sequential context-free grammar.*

Proof. The construction is equivalent to the construction in [5] for g-equivalence. Let $G = (V, \Sigma, P, S)$. A symbol Y is reachable from a symbol X if $X \overset{*}{\Rightarrow} uYv$ for any u, v in V^*; thus every symbol is reachable from itself. For each nonterminal X, let $R(X) = \{Y \in V - \Sigma \mid Y \text{ is reachable from } X \text{ and } X \text{ is reachable from } Y\}$. If $R(X) = \{X\}$ for each X, G is already sequential and we are done. Otherwise, let $[R(X)]$ be a new symbol for each distinct set $R(X)$ and let I be the set of all such new symbols. Let h be the homomorphism on V^* defined by $h(a) = a$ for a in Σ and $h(X) = [R(X)]$ for each X in $V - \Sigma$. Extend h to productions by $h(X \to u) = [R(X)] \to h(u)$. Let $h(P) = \{h(p) \mid p \text{ in } P\}$ and $G_1 = (I \cup \Sigma, \Sigma, h(P), [R(S)])$. Clearly G_1 is sequential and G is in $\mathscr{G}(G_1)$, so $G \leqslant^c G_1$. For each complete interpretation σ of G_1 there is a complete interpretation σ' of G such that $\sigma(G_1)$ can be obtained from $\sigma'(G)$ just as G_1 was obtained from G. So it suffices to show that CONTROL($\{G_1\}$, $\mathscr{L}$) $\subseteq$ CONTROL($\{G\}$, $\mathscr{L}$) for every full semiAFL $\mathscr{L}$.

Using Propositions 2.1 and 2.3 we see that G is c-equivalent to the grammar $G_2 = (V, \Sigma, P', S)$ where $P' = P \cup \{X \to Y \mid Y \in R(X), X \neq Y\}$. Label each rule $X \to Y$ by $[X, Y]$.

Define a finite substitution τ on $h(P)$ by

$$\tau(p) = \{X \to u \mid p = h(X \to u), X \to u \text{ is in } P\}$$
$$\cup \{[Y, X](X \to u) \mid Y \in R(X), Y \neq X, p = h(X \to u), X \to u \text{ is in } P\}.$$

Then for any control set C, $L(G_1, C) = L(G_2, \tau(C))$ which is in CONTROL($\mathscr{G}(G_2)$, $\mathscr{M}(C)$) = CONTROL($\mathscr{G}(G)$, $\mathscr{M}(C)$). ∎

3. Closure Properties

For various well-known families of grammars $\mathscr{G}$ (e.g. context-free grammars, monotonic context-free grammars, regular grammars, context-sensitive grammars) it was shown that CONTROL($\mathscr{G}$, $\mathscr{L}$) is a semiAFL for every semiAFL $\mathscr{L}$ and in many cases a full semiAFL [11]. We would like to establish a similar result for arbitrary grammatical families $\mathscr{G}(G)$. If G is trivial. CONTROL($\mathscr{G}(G)$, $\mathscr{L}$) contains only finite sets and so is not a semiAFL. We strongly conjecture that CONTROL($\mathscr{G}(G)$, $\mathscr{L}$) is a full semiAFL whenever G is nontrivial and $\mathscr{L}$ is a semiAFL. However, in order to prove CONTROL ($\mathscr{G}(G)$, $\mathscr{L}$) closed under inverse homomorphism we have had to put additional conditions on G. One particular condition that is sufficient, but far from necessary, is that G be left derivation bounded.

We can establish closure under homomorphism, intersection with regular sets, and union for CONTROL($\mathscr{G}(G)$, $\mathscr{L}$) whenever $\mathscr{L}$ is closed under e-free finite substitutions and union. To do so it suffices to consider closure under e-input free a-transducer mappings and under union.

Lemma 3.1. *If $\mathscr{L}$ is closed under e-free finite substitutions then* CONTROL($\mathscr{G}(G)$, $\mathscr{L}$) *is closed under e-input free a-transducer mappings. If $\mathscr{L}$ is also closed under union, then* CONTROL($\mathscr{G}(G)$, $\mathscr{L}$) *is also closed under union.*

Proof. Let $\mathscr{L}$ be a family of languages closed under e-free finite substitutions. We shall show that for any grammar G, any e-input free a-transducer M and any control set C in $\mathscr{L}$, $M(L(G, C))$ is in CONTROL($\mathscr{G}(G)$, $\mathscr{L}$). This suffices, for then if H is in $\mathscr{G}(G)$, C is in $\mathscr{L}$ and M is an e-input free a-transducer mapping, we know using Proposition 2.1 that $M(L(H, C)) \in$ CONTROL($\mathscr{G}(H)$, $\mathscr{L}$) $\subseteq$ CONTROL($\mathscr{G}(G)$, $\mathscr{L}$).

Let $G = (V, \Sigma, P, S)$ be a grammar. Let C be a control set in $\mathscr{L}$. Let $M = (K, \Sigma, \Delta, H, q_0, F)$ be an e-input free a-transducer operating on $L(G, C)$. We shall use a variant of the usual cross-product construction.

Since M is e-input free, for states p and q in K and any word w the set $R(p, w, q) = \{y \mid (p, w, e) \overset{*}{\vdash} (q, e, y)\} =$ {all outputs for input w going from p to q} is finite. Let $V_1 = (K \times (V - \Sigma) \times K) \cup \Sigma$. We construct a complete interpretation τ, a subgrammar $G_1 = (V_1, \Delta, P_1, S_1)$ of $\tau(G)$, and an e-free finite substitution σ such that

$$M(L(G, C)) = L(G_1, \sigma(C)) \in \text{CONTROL}(\mathscr{G}(G), \mathscr{L}).$$

We can assume without loss of generality that $F = \{q_f\}$ for some fixed state q_f in K. Let $S_1 = (q_0, S, q_f)$. For each X in $V - \Sigma$, let $\tau(X) = \{(p, X, q) \mid p, q \in K\}$. Let k be the length of the longest right-hand side of a rule in P and let $r = \text{Max}(\{|u| \mid \exists p, q \in K, w \in \Sigma^*, |w| \leqslant k, u \in R(p, w, q)\})$. For a in Σ let $\tau(a) = \{u \in \Delta^* \mid 0 \leqslant |u| \leqslant r\}$.

Now we construct P_1 and σ. If P contains $\lambda: Z \to w$ for w in Σ^*, let $P(\lambda)$ contain all rules of the form $(p, Z, q) \to u$ for $p, q \in K$, $u \in R(p, w, q)$ and $\sigma(\lambda)$ contain the names of all these rules. If P contains $\lambda: Z \to w_1 Y_1 \cdots w_n Y_n w_{n+1}$, $n \geqslant 1$, $w_1, \ldots, w_{n+1} \in \Sigma^*$, $Y_1, \ldots, Y_n \in V - \Sigma$, then $P(\lambda)$ contains all rules of the form $(p, Z, q) \to u_1(s_1, Y_1, t_1)u_2 \cdots u_n(s_n, Y_n, t_n)u_{n+1}$ for $p, q, s_1, t_1, \ldots, s_n, t_n \in K$, $u_1 \in R(p, w_1, s_1)$, $u_i \in R(t_{i-1}, w_i, s_i)$ for

$2 \leqslant i \leqslant n$ and $u_{n+1} \in R(t_n, w_{n+1}, q)$ and $\sigma(\lambda)$ contains the names of all these rules. Then $P_1 = \bigcup_{\lambda \text{ in } P} P(\lambda)$. Clearly G_1 is a subgrammar of $\tau(G)$ and so in $\mathscr{G}(G)$ and has the required properties.

Now suppose that $\mathscr{L}$ is also closed under union and let $G = (V, \Sigma, P, S)$. It suffices to consider $L = L(\tau_1(G), C_1) \cup L(\tau_2(G), C_2)$ for complete interpretations τ_1 and τ_2 of G. We can assume that the vocabularies and hence also the rule sets of $\tau_1(G)$ and of $\tau_2(G)$ are disjoint, that rules are named by themselves and that C_1 and C_2 are over disjoint vocabularies. Let $\tau_i(G)$ have start symbol S_i, $i = 1, 2$. Let S' be a new symbol. We define another complete interpretation $\tau(G)$ of G with start symbol S' by $\tau(A) = \tau_1(A) \cup \tau_2(A)$ for $A \in V - \{S\}$ and $\tau(S) = \tau_1(S) \cup \tau_2(S) \cup \{S'\}$. We define an e-free finite substitution σ by $\sigma(\lambda) = \{\lambda\}$ if λ is a rule in $\tau_i(G)$, $i = 1, 2$ whose left-hand side is not S_i and $\sigma(\lambda) = \{\lambda, S' \to u\}$ if λ is a rule in $\tau_i(G)$ of the form $S_i \to u$, $i = 1, 2$; elsewhere $\sigma(\lambda) = \phi$. Then $L = L(\tau(G), \sigma(C_1 \cup C_2))$ which is in CONTROL$(\mathscr{G}(G), \mathscr{L})$ since $\mathscr{L}$ is closed under union and under e-free finite substitution. ∎

Closure under e-input free a-transducer mappings implies closure under arbitrary homomorphisms and closure under intersection with regular sets. To obtain closure under inverse homomorphism we need only add closure under certain special types of substitutions we shall call c-substitutions [19].

DEFINITION. Let $L \subseteq \Sigma^*$ and let c be a new symbol. A substitution τ on Σ is a *c-substitution* on L if $\tau(a) = c^*ac^*$ for every a in Σ.

We have been unable to establish closure under c-substitution for arbitrary nontrivial grammars. The obvious constructions involve adding the extra c's using productions like $X \to cX$ or $X \to Xc$, but we do not always have the right to add them to $\mathscr{G}(G)$. Let us first see what happens if we do have such symbols in a grammar.

DEFINITION. A nonterminal X in a grammar G is *partially self-embedding* (*pse*) if $X \overset{*}{\Rightarrow} uXv$ for some nonempty terminal string uv; it is *left self-embedding* (*lse*) if $u \neq e$ and *right self-embedding* (*rse*) if $v \neq e$ and *self-embedding* if $u \neq e \neq v$. A grammar G is *strongly pse* if every nonterminal in G except perhaps the start symbol is pse. A grammar G is *self-embedding* if it is reduced and contains some self-embedding symbol.

It is the pse condition we need for inserting c's everywhere. The general idea is that if G is strongly pse, then whatever nonterminal X happens to be leftmost in a derivation string can have productions $X \to cX$ or $X \to Xc$ applied to it at any time, as often as desired, leaving the nonterminal portion of the derivation string unchanged.

LEMMA 3.2. *If G is nontrivial and strongly* pse *and $\mathscr{L}$ is a* semiAFL, *then* CONTROL $(\mathscr{G}(G), \mathscr{L})$ *is closed under c-substitution.*

Proof. We can assume that G is reduced. By Proposition 2.3 we can assume that for some fixed terminal symbol a, G contains a rule $Y \to a$ for every nonterminal Y, a rule $Y \to aY$ for every lse nonterminal Y, and a rule $Y \to Ya$ for every rse nonterminal Y. Now $\tau(u)$ is regular for any c-substitution τ and any finite string u, so by Lemma 3.1

we can assume that G does not contain any rules of the form $S \to u$, for u a terminal string, and S the start symbol of G (otherwise we can add $\tau(u)$ later).

Let $G_1 = (V, \Sigma, P, S_1)$ be an arbitrary member of $\mathscr{G}(G)$ and C an arbitrary member of $\mathscr{L}$. Let τ be a c-substitution on $L(G_1, C)$ for c a symbol not in Σ. We want to find G_2 in $\mathscr{G}(G)$ and C' in $\mathscr{L}$ such that $\tau(L(G_1, C)) = L(G_2, C')$. To simplify notation we shall construct G_2 as a member of $\mathscr{G}(G_1)$ rather than of $\mathscr{G}(G)$. We can assume that G_1 is reduced. For some a in Σ, we can assume that G_1 contains $Y \to a$ for every $Y \in V - \Sigma$, that G_1 contains $Y \to aY$ if Y is obtained from a nonterminal which is lse in G and that G_1 contains $Y \to Ya$ if Y is obtained from a nonterminal which is rse in G. The point is that G_1 as a member of $\mathscr{G}(G)$ is entitled to such rules and if it does not happen to contain these particular ones they could be added with labels which appear nowhere in C and so cannot affect $L(G_1, C)$. Further, every nonterminal except perhaps S_1 must come from a nonterminal which is pse in G. Also G_1 cannot contain a rule $S_1 \to u$ for u in Σ^*, since G cannot have such a rule. Let $N = \#(V - \Sigma)$ and let t be the length of the longest right-hand side of any rule in P. For any Y in $V - \Sigma$, let $f(Y)$ contain all nonterminals of the form

$$(u, Y, v)$$

for u, v in $(\Sigma \cup \{c\}) \cup \{\phi\}$ (ϕ is considered some arbitrary new symbol) and $0 \leqslant |u|$, $|v| \leqslant 2(N+1)(t+2)$. For $a \in \Sigma$, let $f(a) = \Sigma \cup \{c\} \cup \{e\}$.

Again, to simplify notation we shall let rules in G_2 name themselves. We construct G_2 as a subgrammar of $f(G_1)$.

We shall construct C' as $C' = \sigma(C)$ for an e-free regular substitution σ. Now we construct the rule set of G_2 and σ together. Let

$$V_2 = \Sigma \cup \{c\} \bigcup_{Y \in V - \Sigma} f(Y).$$

When we discuss "all legal rules of form..." we imply that all nonterminals appearing are in V_2; this assumption will make our rule sets finite.

First, if Y is lse, let $P(Y, L)$ be the set of all legal rules of the forms

$$\begin{aligned}
&(cu, Y, v) \to c(cu, Y, v),\\
&(cu, Y, v) \to (u, Y, v),\\
&(bu, Y, v) \to b(u, Y, v) \qquad \text{for } b \text{ in } \Sigma,\\
&(e, Y, \phi) \to e.
\end{aligned}$$

If Y is not lse, then $P(Y, L) = \phi$. Similarly, if Y is rse, $P(Y, R)$ is the set of all legal rules of the forms

$$\begin{aligned}
&(u, Y, vc) \to (u, Y, vc)c,\\
&(u, Y, vc) \to (u, Y, v),\\
&(u, Y, vb) \to (u, Y, v)b \qquad \text{for } b \text{ in } \Sigma,\\
&(\phi, Y, e) \to e,
\end{aligned}$$

and if Y is not rse, $P(Y, R) = \phi$. Notice that for $Y \neq S_1$, $P(Y, L) \cup P(Y, R) \neq \phi$.

For a rule λ: $Y \to e$, let P_λ contain all legal rules of the forms

$$(e, Y, v) \to (v, Y, \phi) \qquad \text{for } v \neq \phi \text{ and } Y \text{ lse},$$
$$(u, Y, e) \to (\phi, Y, u) \qquad \text{for } u \neq \phi \text{ and } Y \text{ rse}.$$

Notice that if $Y \to e$ is in P, then $Y \neq S_1$ and $P_\lambda \neq \phi$. Define in this case

$$\sigma(\lambda) = (P(Y, L) \cup P(Y, R))^* P_\lambda (P(Y, L) \cup P(Y, R))^+.$$

For a rule λ: $Y \to a_1 \cdots a_m$, $m \geqslant 1$, each a_i in Σ, let P_λ contain all legal rules of the forms

$$(e, Y, v) \to (ca_1ca_2 \cdots ca_mcv, Y, \phi) \qquad \text{for } v \neq \phi \text{ and } Y \text{ lse},$$
$$(u, Y, e) \to (\phi, Y, uca_1ca_2 \cdots ca_mc) \qquad \text{for } u \neq \phi \text{ and } Y \text{ rse}.$$

Define in this case

$$\sigma(\lambda) = (P(Y, L) \cup P(Y, R))^* P_\lambda (P(Y, L) \cup P(Y, R))^+.$$

Again, $Y \neq S_1$ and $P_\lambda \neq \phi$.

For a rule λ: $X \to u_1X_1 \cdots u_mX_mu_{m+1}$, $m \geqslant 1$, each u_i in Σ^* and X_i in $V - \Sigma$ let P_λ contain all legal rules of the form

$$(u, Y, v) \to (uu_1', X_1, e)(u_2', X_2, e) \cdots (u_{m-1}', X_{m-1}, e)(u_m', X_m, u_{m+1}'v) \quad \text{for } u \neq \phi \neq v$$

where

$$u_i' = e \qquad \text{if} \quad u_i = e$$

and

$$u_i' = ca_1ca_2 \cdots ca_rc \qquad \text{if } u_i = a_1 \cdots a_r,\ r \geqslant 1, \text{ each } a_i \text{ in } \Sigma.$$

Define for this case

$$\sigma(\lambda) = (P(Y, L) \cup P(Y, R))^* P_\lambda .$$

Let

$$P_2 = \bigcup_{\lambda \in P} P_\lambda \cup \bigcup_{Y \in V - \Sigma} (P(Y, L) \cup P(Y, R))$$

and

$$G_2 = (V_2, \Sigma \cup \{c\}, P_2, (c, S_1, c)).$$

Every derivation in G_2 controlled by a member of $C' = \sigma(C)$ simulates a derivation in G_1 controlled by a member of C with the possible generation of additional c's. Thus clearly $L(G_2, C') \subseteq \tau(L(G_1, C))$. To see that $L(G_2, C')$ produces all of $\tau(L(G_1, C))$ notice that whenever G_2 does simulate a complete derivation of G_1 it certainly gets the chance to insert arbitrarily many c's before and after all terminals. Thus we need only argue that $L(G_1, C) \subseteq L(G_2, C')$. The only thing that could "go wrong" in simulating a derivation in G_1 would be for "too many" terminals to pile up in the first or third component of a nonterminal in G_2 and thus cause the derivation to require illegal non-

terminals (not in V_2) and block. Notice that whenever the leftmost nonterminal is lse it has the opportunity to "unload" all of its first component and whenever it is rse it can unload all of its third component. When an ad hoc rule is encountered (simulating $Y \to u$, u a string of terminals) then both components can be unloaded. So the only way terminals can pile up in the first component is for nonterminals to appear which are not lse and yet produce new terminals to the left. At most N such nonterminals can appear or else one of them would be lse. Hence the first component cannot pile up beyond $N(t + 2)$ before encountering either an ad hoc rule or an lse symbol. Similar considerations apply to the third component and rse nonterminals. Hence we can always simulate the appropriate derivation of G_1. ∎

Putting the lemmas together we obtain the following result.

THEOREM 3.3. *If G is nontrivial and strongly* pse, *then for every* semiAFL $\mathscr{L}$, CONTROL$(\mathscr{G}(G), \mathscr{L})$ *is a full* semiAFL.

If a grammar G is ldb we can show that it is c-equivalent to a strongly pse ldb grammar.

LEMMA 3.4. *If G is an* ldb *grammar it is c-equivalent to a strongly* pse *monotonic* ldb *grammar.*

Proof. We can assume that G is reduced and by Theorem 2.9 we can assume that it is monotonic. We proceed by induction on the number of nonterminals in G. If G contains one nonterminal it is the start symbol and so G is trivially strongly pse. Suppose we have shown the desired result for such grammars with at most n nonterminals and G has $n + 1$ nonterminals. Suppose X is a nonterminal in G which is not the start symbol and which is not pse. We cannot have $X \Rightarrow^+ uXv$ because such a derivation would imply $X \overset{*}{\Rightarrow} u'Xv'$ for some nonempty terminal string $u'v'$, since G is reduced and monotonic. Suppose the set of X-rules is $\{X \to x_1, ..., X \to x_m\}$; clearly X does not appear in any x_i. Let τ be the substitution defined by $\tau(X) = \{x_1, ..., x_m\}$ and $\tau(A) = \{A\}$ elsewhere. For a production $Z \to u$ in G, with $Z \neq X$ let $\tau(Z \to u) = \{Z \to v \mid v \text{ is in } \tau(u)\}$; let $\tau(X \to u) = \phi$. Let G' be the grammar with production set $\bigcup_{p \text{ in } G} \tau(p)$. By repeated applications of Proposition 2.8 we can show that G' is c-equivalent to G; obviously G' is ldb and monotonic. Now X does not appear on the right-hand side of any rule of P and is not the start symbol, so we may as well eliminate X from G'. Hence G' contains only n nonterminal symbols and thus is c-equivalent to a monotonic strongly pse ldb grammar by the induction hypothesis. ∎

THEOREM 3.5. *If G is nontrivial and* ldb *then for every* semiAFL $\mathscr{L}$, CONTROL $(\mathscr{G}(G), \mathscr{L})$ *is a full* semiAFL.

When G is not ldb the presence of nontrivial symbols other than the start symbol which are not pse causes difficulties. We cannot use Proposition 2.8 to eliminate such symbols if G is not ldb. However, the results of Section 4 imply that if G is not c-equivalent to any ldb grammar then for any semiAFL $\mathscr{L}$ with LINEARL $\subseteq \mathscr{L} \subseteq$ RE, CONTROL $(\mathscr{G}(G), \mathscr{L}) =$ RE which is a full semiAFL. The characterization theorems of Section 5

imply that if G is c-equivalent to a grammar in Greibach Normal Form or if $\mathscr{L}(G) =$ CFL, then CONTROL($\mathscr{G}(G)$, $\mathscr{L}$) is a full semiAFL for every semiAFL $\mathscr{L}$. Thus it does not look too promising to search for counterexamples. On the other hand, although we conjecture that one can probably force the construction through in all cases, the possible result does not appear worth the effort.

4. Grammars Which are not LDB and RE Sets

In this section we exhibit a series of grammars generating only regular sets without controls yet generating all recursively enumerable sets with linear context-free controls. Our main result will be that for any grammar G, either G is c-equivalent to a left derivation bounded grammar or the G-control operator takes the family of linear context-free languages into the family of recursively enumerable languages, i.e., CONTROL($\mathscr{G}(G)$, LINEARL) = RE. This shows that Theorem 2.11, which says that context-free controls on left derivation bounded grammars yield recursive languages and indeed quasi-realtime languages, is the strongest result of that nature possible, and characterizes grammars c-equivalent to ldb grammars.

We start by showing that for each of the grammars listed below, CONTROL($\mathscr{G}(G)$, LINEARL) = RE. Then we observe that these eight grammars serve to illustrate the general case of a grammar not c-equivalent to any ldb grammar.

The grammars below each contain one partially self-embedding symbol Y and a symbol X which is right expansive in the sense defined below.

Definition. In a grammar G a nonterminal X is *right expansive* (*rex*) if $X \overset{*}{\Rightarrow} uXv$ for strings u and v such that v contains at least one nonterminal. A symbol Y is *reachable from* X if $X \overset{*}{\Rightarrow} xYy$ for any strings x and y (which may be empty); X and Y are *simultaneously reachable from* a symbol W if $W \overset{*}{\Rightarrow} xXyYz$ or $W \overset{*}{\Rightarrow} xYyXz$ for strings x, y, z.

In each of the grammars defined below, the nonterminal vocabulary is $\{S, X, Y, B\}$ with S the initial symbol, and the terminal vocabulary is $\{a\}$. In each case X is a rex symbol with $X \to XB$ a rule, and B is also trivial, generating only the empty string. Symbol Y is pse. The critical point is the relationship between X and Y. Either X is reachable from Y or Y is reachable from X or both are simultaneously reachable from S. We list only the rule sets.

G_{XL}: $S \to X$, $X \to Y$, $X \to XB$, $Y \to aY$, $X \to e$, $Y \to e$, $B \to e$,

G_{XR}: $S \to X$, $X \to Y$, $X \to XB$, $Y \to Ya$, $X \to e$, $Y \to e$, $B \to e$,

G_{YL}: $S \to Y$, $Y \to X$, $X \to XB$, $Y \to aY$, $X \to e$, $Y \to e$, $B \to e$,

G_{YR}: $S \to Y$, $Y \to X$, $X \to XB$, $Y \to Ya$, $X \to e$, $Y \to e$, $B \to e$,

G_{XYL}: $S \to XY$, $X \to XB$, $Y \to aY$, $X \to e$, $Y \to e$, $B \to e$,

G_{XYR}: $S \to XY$, $X \to XB$, $Y \to Ya$, $X \to e$, $Y \to e$, $B \to e$,

G_{YXL}: $S \to YX$, $X \to XB$, $Y \to aY$, $X \to e$, $Y \to e$, $B \to e$,

G_{YXR}: $S \to YX$, $X \to XB$, $Y \to Ya$, $X \to e$, $Y \to e$, $B \to e$.

LEMMA 4.1. *For* $u \in \{XL, XR, YL, YR, XYL, XYR, YXL, YXR\}$

$$\mathscr{L}(G_u) = \text{REGULARL}$$

and

$$\text{CONTROL}(\mathscr{G}(G_u), \text{LINEARL}) = \text{RE}$$

and so $\text{LINEARL} = \mathscr{H}(\mathscr{L}(G_u) \wedge \text{LINEARL}) \subsetneq \text{CONTROL}(\mathscr{G}(G_u), \text{LINEARL})$.

Proof. None of the symbols in any G_u are self-embedding and hence $\mathscr{L}(G_u) = \text{REGULARL}$ [5].

We shall use the representation of recursively enumerable sets by means of general rewriting systems.[7] In each case we shall construct for an arbitrary general rewriting system G a grammar $G_u(G)$ in $\mathscr{G}(G_u)$ and a linear context-free language L_u such that $L(G) = L(G_u(G), L_u)$. Since CONTROL(CFG, CFL) = RE, this will suffice. We shall give the proof for G_{XL} in some detail and merely outline the construction in other cases.

Let $G = (V, \Sigma, P, W)$ be a general rewriting system; we can assume the symbols of V to be distinct from S, X, Y, and B. Let $\mathscr{S}_1$, $\mathscr{S}_2$, ϕ, $\bar{\phi}$, and $\hat{\phi}$ be new symbols and let $\bar{h}$ and $\hat{h}$ be homomorphisms such that $\bar{h}$ maps any symbol A in V into a new symbol $\bar{A}$ and $\hat{h}$ maps A into another new symbol $\hat{A}$. The grammar $G_{XL}(G)$ will have nonterminal vocabulary $\bar{h}(V) \cup \{S, X, Y, \bar{\phi}\}$ and terminal vocabulary Σ. It is a subgrammar of $\tau(G_{XL})$ for a complete interpretation τ with $\tau(a) = \Sigma$, $\tau(S) = \{S\}$, $\tau(X) = \{X\}$, $\tau(Y) = \{Y\}$, and $\tau(B) = \bar{h}(V) \cup \{\phi\}$. For clarity we list below the productions of G_{XL} and their labels.

Label	Production	
A	$Y \to AY$	for all A in Σ,
$\bar{A}$	$X \to X\bar{A}$	for all A in $V \cup \{\phi\}$,
$\hat{A}$	$\bar{A} \to e$	for all A in $V \cup \{\phi\}$,
$\mathscr{S}_1$	$S \to X$	
$\mathscr{S}_2$	$Y \to e$	
ϕ	$X \to Y$	

The language L_{XL} consists of all and only strings of the form $\mathscr{S}_1\alpha\mathscr{S}_2\beta$ where

$$\alpha = \bar{h}(w_{n-1})\bar{\phi} \cdots \bar{h}(w_1)\,\bar{\phi}\bar{h}(W)\,\bar{\phi}\phi w_n\,,$$
$$\beta = \hat{\phi}\hat{h}(y_1{}^R)\,\hat{\phi}\hat{h}(y_2{}^R) \cdots \hat{\phi}\hat{h}(y_n{}^R),$$

for $n \geqslant 1$, $w_i, y_i \in V^*$, $y_i \Rightarrow_G w_i$ for $1 \leqslant i \leqslant n$, and $w_n \in \Sigma^*$.

The point is that the language $\{w^R \phi y \mid y \Rightarrow w\}$ is linear context-free and by extension so is L_{XL}. To use other concepts, a *single step* of a Turing machine or a grammar can be

[7] A *general rewriting system* is a quadruple $G = (V, \Sigma, P, S)$ where V is a finite vocabulary, $\Sigma \subseteq V$, $S \in V - \Sigma$, and P is a finite set of rules of the form $x \to y$, $x \in V^+ - \Sigma^+$, $y \in V^*$. If $x \to y \in P$ and $u, v \in V^*$ then $uxv \Rightarrow uyv$; $\overset{*}{\Rightarrow}$ is the transitive reflexive closure of $\Rightarrow$. The language generated by G is $L(G) = \{w \in \Sigma^* \mid S \overset{*}{\Rightarrow} w\}$.

effected by a finite state machine. Similar constructions and their justification can be found in Refs. [1, 9, 16], and elsewhere.

Notice that for any v in V^*, the word derived from X under control word $\bar{h}(v)$ is $X\bar{h}(v^R)$. Suppose we have α and β as described above Applying control word $\bar{h}(w_{n-1})\bar{\phi} \cdots \bar{h}(w_1)\,\bar{\phi}\bar{h}(W)\bar{\phi}$ to S yields $X\bar{\phi}\bar{h}(W^R)\,\bar{\phi}\bar{h}(w_1{}^R)\bar{\phi} \cdots \bar{\phi}\bar{h}(w_{n-1}^R)$. Then rule ϕ turns X into Y and control word w_n generates w_nY from Y. Hence for $\mathcal{S}_1\alpha\mathcal{S}_2$ we have

$$S \overset{\mathcal{S}_1\alpha\mathcal{S}_2}{\Rightarrow} w_n\bar{\phi}\bar{h}(W^R)\,\bar{\phi}\bar{h}(w_1{}^R)\bar{\phi} \cdots \bar{\phi}\bar{h}(w_{n-1}^R).$$

If we try to apply control word β to the string produced so far, we see that first we must apply $\hat{\phi}\hat{h}(\,y_1{}^R)\hat{\phi}$ which can only serve to erase $\bar{\phi}\bar{h}(\,y_1{}^R)\bar{\phi}$. Since $\bar{\phi}$ is distinct from any symbol in $\bar{h}(V)$, we must have $\bar{h}(\,y_1{}^R) = \bar{h}(W^R)$ and thus $W = y_1$. If this holds, then we have produced from S, $w_n\bar{\phi}\bar{h}(w_1{}^R)\bar{\phi} \cdots \bar{\phi}\bar{h}(w_{n-1}^R)$. Continuing in this fashion, we see that for β to apply, we must have $W = y_1$ and $w_i = y_{i+1}$ for $1 \leqslant i \leqslant n-1$. If this relationship holds, then

$$S \overset{\mathcal{S}_1\alpha\mathcal{S}_2\beta}{\Rightarrow} w_n\,.$$

Now if $\mathcal{S}_1\alpha\mathcal{S}_2\beta$ is in L_{XL}, then $y_i \Rightarrow_G w_i$ for $1 \leqslant i \leqslant n$, and w_n in Σ^*. Combining these two sets of conditions we have

$$W = y_1 \underset{G}{\Rightarrow} w_1 = y_2 \underset{G}{\Rightarrow} \cdots \underset{G}{\Rightarrow} w_{n-1} = y_n \underset{G}{\Rightarrow} w_n$$

and so w_n is in $L(G)$.

Thus we have shown that $L(G_{XL}(G), L_{XL}) \subseteq L(G)$. Reversing the argument, we note that if w is in $L(G)$, then for appropriate $n \geqslant 1$, $w_1, \ldots, w_{n-1} \in V^*$, and $w_n = w$, we have $W \Rightarrow_G w_1 \Rightarrow_G \cdots \Rightarrow_G w_{n-1} \Rightarrow_G w_n$ and so if $\alpha = \bar{h}(w_{n-1})\bar{\phi} \cdots \bar{h}(w_1)\,\bar{\phi}\bar{h}(W)\,\bar{\phi}\phi w_n$ and

$$\beta = \hat{\phi}\hat{h}(W^R)\,\hat{\phi}\hat{h}(w_1{}^R) \cdots \hat{\phi}\hat{h}(w_{n-1}^R)$$

then $\mathcal{S}_1\alpha\mathcal{S}_2\beta$ is in L_{XL} and $S \overset{\mathcal{S}_1\alpha\mathcal{S}_2\beta}{\Rightarrow} w_n$. Hence $L(G) \subseteq L(G_{XL}(G), L_{XL})$.

The proof for the other seven cases is similar. We shall only indicate how each $G_u(G)$ differs from $G_{XL}(G)$ and how L_u differs from L_{XL}. In each case we assume that L_u contains all and only words of the form $\mathcal{S}_1\alpha\mathcal{S}_2\beta$ where α and β are expressed in terms of words $W, w_1, \ldots, w_n, y_1, \ldots, y_n$ with $n \geqslant 1$, $w_i, y_i \in V^*$, and $y_i \Rightarrow_G w_i$ for $1 \leqslant i \leqslant n$, and w_n in Σ^*. Thus we need only describe the format of α and β (see Table I). ∎

Now we observe that the situation above is completely general. Whenever G contains a partially self-embedding symbol and a right expansive symbol related in one of the four ways above, then $\mathcal{G}(G)$ can generate all recursively enumerable languages with only linear controls.

LEMMA 4.2. *If a reduced grammar G contains a partially self-embedding symbol Y and a right expansive symbol X, such that either Y is reachable from X or X is reachable from Y or both X and Y are simultaneously reachable from the start symbol, then*

$$\text{CONTROL}(\mathcal{G}(G), \text{LINEARL}) = \text{RE}.$$

TABLE I

Grammar	Production change	α	β
XR	A labels $Y \to YA$ instead of $Y \to AY$	$\bar{h}(W^R)\bar{\phi}\bar{h}(w_1{}^R)\cdots\bar{\phi}\bar{h}(w_{n-1}^R)\bar{\phi}\phi w_n{}^R$	$\hat{\phi}\hat{h}(y_n)\cdots\hat{\phi}\hat{h}(y_1)$
YL	$\mathscr{S}_1$ labels $S \to Y$ $\mathscr{S}_2$ labels $X \to e$ ϕ labels $Y \to X$	$w_n\phi\bar{h}(w_{n-1})\bar{\phi}\cdots\bar{h}(w_1)\bar{\phi}\bar{h}(W)\bar{\phi}$	$\hat{\phi}\hat{h}(y_1{}^R)\cdots\hat{\phi}\hat{h}(y_n{}^R)$
YR	$\mathscr{S}_1$ labels $S \to Y$ $\mathscr{S}_2$ labels $X \to e$ ϕ labels $Y \to X$ A labels $Y \to YA$	$w_n{}^R\phi\bar{\phi}\bar{h}(w_{n-1}^R)\cdots\bar{\phi}\bar{h}(w_1{}^R)\bar{\phi}\bar{h}(W^R)$	$\hat{h}(y_1)\hat{\phi}\cdots\hat{\phi}(y_n)$
XYL	$\mathscr{S}_1$ labels $S \to XY$ $\mathscr{S}_2$ labels $X \to e$ ϕ labels $Y \to e$	$\bar{\phi}\bar{h}(y_n{}^R)\cdots\bar{\phi}\bar{h}(y_1{}^R)$	$\hat{h}(W)\hat{\phi}\hat{h}(w_1)\cdots\hat{\phi}\hat{h}(w_{n-1})\hat{\phi}w_n\phi$
XYR	$\mathscr{S}_1$ labels $S \to XY$ $\mathscr{S}_2$ labels $X \to e$ ϕ labels $Y \to e$ A labels $Y \to YA$	$\bar{\phi}\bar{h}(y_n)\cdots\bar{\phi}\bar{h}(y_1)$	$\hat{h}(W^R)\hat{\phi}\hat{h}(w_1{}^R)\cdots\hat{\phi}\hat{h}(w_{n-1}^R)\hat{\phi}w_n{}^R\phi$
YXL	$\mathscr{S}_1$ labels $S \to YX$ $\mathscr{S}_2$ labels $X \to e$ ϕ labels $Y \to e$	$w_n\phi\bar{h}(w_{n-1})\bar{\phi}\cdots\bar{h}(w_1)\bar{\phi}\bar{h}(W)\bar{\phi}$	$\hat{h}(y_1{}^R)\hat{\phi}\cdots\hat{\phi}\hat{h}(y_n{}^R)$
YXR	$\mathscr{S}_1$ labels $S \to YX$ $\mathscr{S}_2$ labels $X \to e$ ϕ labels $Y \to e$ A labels $Y \to YA$	$w_n{}^R\phi\bar{h}(w_{n-1}^R)\bar{\phi}\cdots\bar{h}(w_1{}^R)\bar{\phi}\bar{h}(W^R)\bar{\phi}$	$\hat{\phi}\hat{h}(y_1)\cdots\hat{\phi}\hat{h}(y_n)$

Proof. It will suffice to show that G is c-equivalent to a grammar G' such that $\mathscr{G}(G')$ contains a grammar H which is one of the eight grammars mentioned in Lemma 4.1 above. The grammar G' will contain the productions of G plus (possibly) certain extra productions of the form $W \to w$ where $W \overset{*}{\Rightarrow} w$ in G, so G' will be c-equivalent to G by Proposition 2.3. The choice of H as one of the G_u depends on whether Y is left self-embedding ($H = G_{vL}$) or right self-embedding ($H = G_{vR}$) and whether Y is reachable from X ($H = G_{Xv}$) or X is reachable from Y ($H = G_{Yv}$) or X and Y are simultaneously reachable from the start symbol S ($H = G_{XYv}$ or $H = G_{YXv}$).

Since X is right expansive and Y is partially self-embedding, we have in G derivations $X \Rightarrow^+ x_1Xx_2Bx_3$, $X \Rightarrow^+ x_4$, $B \Rightarrow^+ x_5$, $Y \Rightarrow^+ y_1Yy_2$, $Y \Rightarrow^+ y_3$ for appropriate terminal strings x_i, $1 \leqslant i \leqslant 5$ and y_j, $1 \leqslant j \leqslant 3$, with $y_1y_2 \neq e$, and a nonterminal B. Then G' contains productions $X \to x_1Xx_2Bx_3$, $X \to x_4$, $B \to x_5$, $Y \to y_1Yy_2$, and $Y \to y_3$. Any of the symbols S, X, Y, and B may be equal to each other in G and G'

but can be given distinct names in H, an interpretation of G'. So H will contain productions $X \to XB$, $X \to e$, $B \to e$, and $Y \to e$ and either $Y \to aY$ or $Y \to Ya$ for $y_1 \neq e$ or $y_2 \neq e$, respectively.

If X and Y are simultaneously reachable from S, then G contains either $S \Rightarrow^+ u_1Xu_2Yu_3$ or $S \Rightarrow^+ u_1Yu_2Xu_3$ for appropriate terminal strings u_1, u_2, and u_3, so G' contains $S \to u_1Xu_2Yu_3$ or $S \to u_1Yu_2Xu_3$ and H contains $S \to XY$ or $S \to YX$, accordingly. Thus in this case H is either G_{XYv} or G_{YXv} for v in $\{L, R\}$, and we are done.

Otherwise we have in G either $X = Y$ or $X \Rightarrow^+ u_1Yu_2$ or $Y \Rightarrow^+ u_1Xu_2$ for appropriate terminal strings u_1 and u_2. If $X = Y$, then $Y \Rightarrow^+ y_1Xy_2$ so we can take $y_1 = u_1$ and $y_2 = u_2$. Hence we have in G' either $X \to u_1Yu_2$ or $Y \to u_1Xu_2$ and in H either $X \to Y$ or $Y \to X$. Consider the case $X \Rightarrow^+ u_1Yu_2$. Either $X = S$ or $S \Rightarrow^+ w_1Xw_2$ in G for terminal strings w_1 and w_2. If $S = X$, then $S \Rightarrow^+ x_1Xx_2Bx_3 \Rightarrow^+ x_1Xx_2x_5x_3$ so we can take $w_1 = x_1$ and $w_2 = x_2x_5x_3$. Hence G' contains $S \to w_1Xw_2$ and H contains $S \to X$. Thus in this case H is either G_{XL} or G_{XR}. A similar argument in the case $Y \Rightarrow^+ u_1Xu_2$ shows that H can contain $S \to Y$ and so be equal to G_{YL} or G_{YR}.

Thus in all cases we have H equal to one of the G_u and RE = CONTROL($\mathscr{G}(H)$, LINEARL) $\subseteq$ CONTROL($\mathscr{G}(G')$, LINEARL) = CONTROL($\mathscr{G}(G)$, LINEARL) $\subseteq$ RE. ∎

We have the following useful corollary.

COROLLARY 4.3. *If G is a reduced grammar with a nontrivial symbol which is not* ldb, *then* CONTROL($\mathscr{G}(G)$, LINEARL) = RE.

We now observe that either one of the situations described in Lemma 4.2 occurs for grammar G or else G is c-equivalent to an ldb subgrammar.

THEOREM 4.4. *Let G be a reduced grammar. Either* CONTROL($\mathscr{G}(G)$, LINEARL) = RE *or G is c-equivalent to an* ldb *grammar. The two cases are mutually exclusive and in the second case G is c-equivalent to an* ldb *subgrammar.*

Proof. First suppose that G is trivial. There are three cases, $L(G) = \phi$, $L(G) = \{e\}$, and $L(G)$ is finite but contains a nonempty word. In the first case, G is the empty grammar since G is reduced, and so is itself ldb. In the second case, every nonterminal symbol must generate the empty string and for any nontrivial $\mathscr{L}$, CONTROL($\mathscr{G}(G)$, $\mathscr{L}$) = $\{\phi, \{e\}\}$. Let I_1 contain all symbols X such that $X \to e$ is in G and put $X \to e$ in P_1. For each $i \geqslant 1$, let I_{i+1} contain all symbols X which are not in any I_j, $j \leqslant i$ but for which G contains a rule $X \to Y_1 \cdots Y_k$ with each Y_r in some I_j, $j \leqslant i$ and place $X \to Y_1 \cdots Y_k$ in P_i. If G has n nonterminals, let $P' = \bigcup_{1 \leqslant i \leqslant n} P_i$. Clearly each nonterminal is in some I_i, $1 \leqslant i \leqslant n$ including the start symbol of G. The subgrammar G' of G with production set P' can have no derivation of the form $X \Rightarrow^+ uXv$ and so is a fortiori ldb; we have $L(G') = \{e\}$ and so G' is trivially c-equivalent to G.

Now consider the case where $L(G)$ is finite and contains at least one nonempty word. Then $\mathscr{L}(G)$ is the family of finite sets as is CONTROL($\mathscr{G}(G)$, $\mathscr{L}$) for any semiAFL $\mathscr{L}$. We use a construction similar to the one above. Here we start with P_1 containing all ad hoc productions $X \to w$ with w a terminal string and X in the corresponding set I_1.

Then for $i \geqslant 1$, if X is not in any I_j for $j \leqslant i$ but there is a rule $X \to A_1 \cdots A_r$ with each A_k either a terminal or in some I_j, $j \leqslant i$, then X is placed in I_{i+1} and $X \to A_1 \cdots A_r$ in P_i. As before, if G has n nonterminals, every nonterminal is in some I_k, for $1 \leqslant i \leqslant n$, including the start symbol of G. The subgrammar G' of G with production set $P' = \bigcup_{1 \leqslant i \leqslant n} P_i$ can have no derivation $X \Rightarrow^+ uXv$ and so is a fortiori ldb. Since $L(G')$ is finite and contains some nonempty word, G is c-equivalent to G'.

Now suppose G is nontrivial, and so contains at least one pse symbol; let $G = (V, \Sigma, P, S)$. If G satisfies the hypotheses of Lemma 4.2, then CONTROL($\mathscr{G}(G)$, LINEARL) = RE and we are done. Otherwise G has the following property which is inherited by any grammar in $\mathscr{G}(G)$.

(*) If a derivation of a terminal string from S uses a symbol which is pse it cannot use any symbol which is rex and if such a derivation uses a symbol which is rex it cannot use any symbol which is pse.

If G has no rex symbols it is already ldb. Assume G has at least one rex symbol.

Let I_1 contain Σ plus all nonterminals which are not rex, I_2 contain Σ plus all nonterminals which are not pse, P_1 contain all productions in P involving only members of I_1, and P_2 contain all productions in P involving only members of I_2. Let $G_1 = (I_1, \Sigma, P_1, S)$ and $G_2 = (I_2, \Sigma, P_2, S)$. Note that S must be in both I_1 and I_2 since if S were either rex or pse, G would satisfy the hypotheses of Lemma 4.2. Since I_1 has no rex symbols, G_1 is an ldb subgrammar of G and is nontrivial since S is nontrivial. Condition (*) ensures that any completed derivation involving a rex symbol takes place entirely within G_2; since I_2 has no pse symbols, $L(G_2)$ is finite.

We claim that G is c-equivalent to G_1. Since G_1 is a subgrammar of G, $G_1 \leqslant^c G$. It remains to show that $G \leqslant^c G_1$.

Now let H be in $\mathscr{G}(G)$ and let C be any control set. Since (*) is inherited by members of $\mathscr{G}(G)$, we can divide H as we divided G into subgrammars H_1 in $\mathscr{G}(G_1)$ and H_2 in $\mathscr{G}(G_2)$ such that every derivation of H takes place either entirely within H_1 or entirely within H_2 and $L(H_2)$ is finite. Then $L(H, C) = L(H_1, C) \cup L(H_2, C)$ and $L(H_2, C)$ is finite. Let $L(H_2, C) = \{w_1, \ldots, w_n\}$. Let S_H be the start symbol of H. Let $S \Rightarrow u_1 \Rightarrow \cdots \Rightarrow u$ be some derivation of a nonempty terminal string in G_1. We can form H_1' in $\mathscr{G}(G_1)$ by adding to H_1, n new interpretations of the productions in this derivation using all new names for the nonterminals in such a way that for each i, $1 \leqslant i \leqslant n$, $S_H \Rightarrow^{\lambda_i} w_i$ for a control word λ_i not in C and the names of the added productions do not appear anywhere in C. Thus for $C' = C \cup \{\lambda_1, \ldots, \lambda_n\}$, $L(H_1', C') = L(H_1, C) \cup \{w_1, \ldots, w_n\} = L(H, C)$ and clearly C' is in any semiAFL containing C. ∎

COROLLARY 4.5. *For any* semiAFL $\mathscr{L}$ *such that* LINEARL $\subseteq \mathscr{L} \subseteq$ RE, *and any nontrivial grammar* G, CONTROL($\mathscr{G}(G)$, $\mathscr{L}$) *is full* semiAFL.

COROLLARY 4.6. *A grammar* G *is c-equivalent to a left derivation bounded grammar if and only if* CONTROL($\mathscr{L}(G)$, LINEARL) $\neq$ RE *if and only if* CONTROL($\mathscr{G}(G)$, LINEARL) $\subsetneq$ QUASI.

5. Characterizations for Grammars Which are not LDB

Ginsburg and Spanier proved [11] that for the whole class of context-free grammars and any semiAFL $\mathscr{L}$

$$(*)\quad \text{CONTROL(CFG}, \mathscr{L}) = \hat{\mathscr{H}}(\mathscr{L}(\text{CFG}) \wedge \mathscr{L}) = \hat{\mathscr{H}}(\text{CFL} \wedge \mathscr{L}).$$

A similar result holds for the family of regular grammars where we have [11] for every semiAFL $\mathscr{L}$

$$\begin{aligned}\text{CONTROL(REGULARG}, \mathscr{L}) &= \hat{\mathscr{H}}(\mathscr{L}(\text{REGULARG}) \wedge \mathscr{L}) \\ &= \hat{\mathscr{H}}(\text{REGULARL} \wedge \mathscr{L}) = \hat{\mathscr{H}}(\mathscr{L}).\end{aligned}$$

It is natural to ask whether the analog of (*) holds for every grammatical family. In particular, when does a context-free grammar have property (**) below?

(**) For every full semiAFL $\mathscr{L}$,

$$\text{CONTROL}(\mathscr{G}(G), \mathscr{L}) = \hat{\mathscr{H}}(\mathscr{L}(G) \wedge \mathscr{L}).$$

We shall see that (**) holds only under very restricted circumstances. It is possible to have CONTROL$(\mathscr{G}(G), \mathscr{L})$ properly contained in $\hat{\mathscr{H}}(\mathscr{L}(G) \wedge \mathscr{L})$ or $\hat{\mathscr{H}}(\mathscr{L}(G) \wedge \mathscr{L})$ properly contained in CONTROL$(\mathscr{G}(G), \mathscr{L})$ or to have the two families incomparable, neither contained in the other.

Let us give a simple example. Let G be the grammar $(\{S, a\}, \{a\}, \{S \to aS, S \to a\}, S)$ and G_R the grammar $(\{S, a\}, \{a\}, \{S \to Sa, S \to a\}, S)$. If $H = (V, \Sigma, P, S)$ is in $\mathscr{G}(G)$, obviously $\mathscr{G}(G_R)$ contains $H_R = (V, \Sigma, P^R, S)$ where $P^R = \{Z \to u^R \mid Z \to u \in P\}$ and for any control set C, $L(H_R, C) = (L(H, C))^R$. Now $\mathscr{G}(G)$ is the family of regular grammars. Thus for every full semiAFL $\mathscr{L}$, CONTROL $(\mathscr{G}(G), \mathscr{L}) = \mathscr{L}$ but CONTROL$(\mathscr{G}(G_R), \mathscr{L}) = \mathscr{L}^R$. Now let $\mathscr{L}$ be a full semiAFL not closed under reversal; for example, $\mathscr{L}$ could be the full semiAFL generated by $L = \{a^n b^m \mid 0 \leqslant n \leqslant m\}$ [7]. Then CONTROL$(\mathscr{G}(G_R), \mathscr{L}) = \mathscr{L}^R$ while $\hat{\mathscr{H}}(\mathscr{L}(G_R) \wedge \mathscr{L}) = \hat{\mathscr{H}}(\text{REGULARL} \wedge \mathscr{L}) = \mathscr{L}$ and $\mathscr{L}$ and $\mathscr{L}^R$ are incomparable.

This argument can be extended to show that (**) holds for a left derivation bounded grammar G if and only if G is c-equivalent to a regular grammar. If G is trivial, G is always c-equivalent to a regular grammar. Suppose G is reduced, nontrivial, and ldb and that (**) holds for G. If G is self-embedding, then $\mathscr{L}(G)$ contains LINEARL [5] and so $\hat{\mathscr{H}}(\mathscr{L}(G) \wedge \text{LINEARL}) = \text{RE}$ [1]. But then (**) would imply CONTROL $(\mathscr{G}(G), \text{LINEARL}) = \text{RE}$, contradicting Theorem 2.11. Thus G cannot be self-embedding. Hence $\mathscr{L}(G) = \text{REGULARL}$ and thus (**) would mean that CONTROL $(\mathscr{G}(G), \mathscr{L}) = \mathscr{L}$ for every full semiAFL $\mathscr{L}$, so G is c-equivalent to the regular grammar $G = (\{S, a\}, \{a\}, \{S \to aS, S \to a\}, S)$ mentioned above. Also note that G contains no right self-embedding symbols or else by the proof of Lemma 2.12, CONTROL $(\mathscr{G}(G), \mathscr{L})$ contains $\mathscr{L}^R$ for any full semiAFL $\mathscr{L}$, impossible for $\mathscr{L}$ not closed under reversal. Hence we have shown the following theorem.

THEOREM 5.1. *Let G be a reduced left derivation bounded grammar. If*

$$\text{CONTROL}(\mathscr{G}(G), \mathscr{L}) = \hat{\mathscr{H}}(\mathscr{L}(G) \wedge \mathscr{L})$$

for every full semiAFL $\mathscr{L}$, *then G contains no right self-embedding symbols and is c-equivalent to a regular grammar.*

Thus the characterization (**) can hold only for grammars that are not left derivation bounded except for the case already studied, that of regular grammars.

We shall give conditions under which the two possible inclusions in (**) hold. These conditions will turn out to be closely related to two well-known normal forms for context-free grammars, Chomsky Normal Form [4] and Greibach Normal Form [16]. These forms are normal forms for g-equivalence but we shall see that they are not normal forms for c-equivalence and involve independent conditions on a grammar. Our two main results in this section will be that (**) holds for an arbitrary grammar G if and only if G is c-equivalent to some grammar in Greibach Normal Form and that in particular (**) always holds if $\mathscr{L}(G) =$ CFL.

First we shall see that a weak form of Chomsky Normal Form suffices to establish the inclusion $\hat{\mathscr{H}}(\mathscr{L}(G) \wedge \mathscr{L}) \subseteq \text{CONTROL}(\mathscr{G}(G), \mathscr{L})$ for all semiAFLs $\mathscr{L}$.

DEFINITION. A grammar G is in *weak Chomsky Normal Form* (wCNF) if for every rule $Z \to Ay$ in P with A in V, y is in $(V - \Sigma)^*$. It is in *Chomsky Normal Form* (CNF) if every rule is in the forms $Z \to XY$, $Z \to a$ and $S \to e$ for X, Y in $V - \Sigma - \{S\}$ and a in Σ.

Every grammar is g-equivalent to one in CNF [5] but we shall see that there are grammars which are not c-equivalent to any grammar in wCNF.

The wCNF property simply states that terminals can appear only leftmost (but need not appear at all) in rules. This property allows one to simulate the intersection of members of $\mathscr{L}(G)$ with members of $\mathscr{L}$.

THEOREM 5.2. *If G is in* wCNF *then for every* semiAFL $\mathscr{L}$,

$$\hat{\mathscr{H}}(\mathscr{L}(G) \wedge \mathscr{L}) \subseteq \text{CONTROL}(\mathscr{G}(G), \mathscr{L}).$$

Proof. In view of Lemma 3.1 it suffices to show that $\mathscr{L}(G) \wedge \mathscr{L}$ is contained in $\text{CONTROL}(\mathscr{G}(G), \mathscr{L})$. Notice that the wCNF property is inheritable in the sense that if G is in wCNF so is every member of $\mathscr{G}(G)$. So it suffices to prove that $L(G) \cap C \in \text{CONTROL}(\mathscr{G}(G), \mathscr{L})$ whenever G is in wCNF, C is in $\mathscr{L}$, and $\mathscr{L}$ is a semiAFL.

Let $G = (V, \Sigma, P, S)$ and let C be in $\mathscr{L}$. We shall construct an e-free regular substitution τ such that $(L(G) \cap C) - \{e\} = L(G, \tau(C))$. For convenience, let the productions in G name themselves and assume these names different from any symbol appearing in C. The idea is to substitute for a word $a_1 \cdots a_n$ in C (a_i individual symbols) all possible control words of the form $x_1\, p(a_1) \cdots x_n\, p(a_n)\, x_{n+1}$ such that the rules in the x_i do not produce new terminal symbols while rule $p(a_i)$ produces a_i. Since G is in wCNF and we are controlling left-to-right derivations, when a_i appears it will be left of all nonterminals

and right of all current terminals. Thus control word $x_1 p(a_1) \cdots x_n p(a_n) x_{n+1}$ will produce $a_1 \cdots a_n$ in grammar G if it produces anything and each word in $L(G)$ can be so generated.

For each terminal symbol a in Σ let $P(a) = \{Z \to ay \mid Z \to ay \in P\}$; let $P' = \{Z \to y \mid Z \to y \in P, y \in (V - \Sigma)^*\}$. Because G is in wCNF, every rule is either in P' or in some $P(a)$. Let $\tau(a) = (P')^* P(a) (P')^*$. The rules in P' will give us the strings x_i while $P(a_i)$ will give us an appropriate $p(a_i)$. If e is not in $L(G) \cap C$, let $C' = \tau(C) \in \mathscr{L}$. Then $L(G) \cap C = L(G, C') \in \text{CONTROL}(\mathscr{G}(G), \mathscr{L})$. If e is in $L(G) \cap C$, let u be any control word in $(P')^+$ such that $S \Rightarrow^u e$ and let $C' = \tau(C) \cup \{u\}$. Then $L(G) \cap C = L(G, C') \in \text{CONTROL}(\mathscr{G}(G), \mathscr{L})$. ∎

Remark. In Theorem 5.2 we really only require that $\mathscr{L}$ be closed under e-free regular substitution and union with unit sets (sets of the form $\{w\}$).

COROLLARY 5.3. *If there is a full* semiAFL $\mathscr{L}$ *such that* $\hat{\mathscr{H}}(\mathscr{L}(G) \wedge \mathscr{L})$ *is not contained in* CONTROL$(\mathscr{G}(G), \mathscr{L})$, *then* G *is not* c*-equivalent to any grammar in* wCNF.

Proof. If $G \equiv^c G_1$ and G_1 is in wCNF,

$$\hat{\mathscr{H}}(\mathscr{L}(G) \wedge \mathscr{L}) = \hat{\mathscr{H}}(\mathscr{L}(G_1) \wedge \mathscr{L}) \subseteq \text{CONTROL}(\mathscr{G}(G_1), \mathscr{L}) = \text{CONTROL}(\mathscr{G}(G), \mathscr{L}),$$

contradicting the hypothesis. ∎

We have already exhibited a grammar G_R such that $\mathscr{L} = \hat{\mathscr{H}}(\mathscr{L}(G_R) \wedge \mathscr{L})$ is not contained in CONTROL$(\mathscr{G}(G_R), \mathscr{L}) = \mathscr{L}^R$ if $\mathscr{L}$ is not closed under reversal. Hence G_R is an example of a very simple grammar not c-equivalent to any grammar in wCNF. Another example is $G_L = (\{S, a\}, \{a\}, \{S \to aSa, S \to a\}, S)$. Clearly $\mathscr{G}(G_L)$ is the family of linear context-free grammars and $\mathscr{L}(G_L)$ is the class of linear context-free languages. In this case $\hat{\mathscr{H}}(\mathscr{L}(G_L) \wedge \mathscr{L}(G_L))$ is the family of recursively enumerable languages [1] but Theorem 2.11(c) tells us that CONTROL$(\mathscr{G}(G_L), \mathscr{L}(G_L))$ is properly contained in QUASI (and indeed considerations in Section 6 tell us that it is properly contained in the family of checking automaton languages (cf. [14, 17, 33])). Thus G_L is not c-equivalent to any grammar in wCNF.

COROLLARY 5.4. *There are linear context-free grammars not* c*-equivalent to any grammar in* wCNF.

We now investigate the other half of (**), the potential inclusion CONTROL $(\mathscr{G}(G), \mathscr{L}) \subseteq \hat{\mathscr{H}}(\mathscr{L}(G) \wedge \mathscr{L})$. It is not difficult to show that $L(G, C)$ is in $\hat{\mathscr{H}}(\text{CFL} \wedge \mathscr{L})$ for C in $\mathscr{L}$. A new grammar G' is formed in which production $Z \to y$ in G labeled p becomes $Z \to py$. Thus the new grammar G' generates words of the form $x_1 w_1 \cdots x_n w_n x_{n+1}$ where G generates $w = w_1 \cdots w_n$ under control word $x = x_1 \cdots x_{n+1}$, and whenever w is generated under x in G some word of this form appears in G'. Now one must ensure that x is in C. This is done by forming C' from C by scattering in the terminals of G and then intersecting $L(G')$ with C'. Finally one must erase the production names to end up with $L(G, C)$. The catch is that one must be able to go from $Z \to y$ to $Z \to py$ without leaving $\mathscr{G}(G)$ and that is not always possible.

One property which makes this legal is a weak form of Greibach Normal Form.

DEFINITION. A nonterminal Z in a grammar G is *left recursive* if $Z \Rightarrow^+ Zv$ for some string v (which may be empty). A grammar is *left recursion free* if it has no left recursive nonterminals. A grammar $G = (V, \Sigma, P, S)$ is in *weak Greibach Normal Form* (*wGNF*) if whenever $Z \to Ay$ is in P for A in V, then A is in Σ and whenever $Z \to e$ is in P then $Z = S$ and S does not appear on the right-hand side of any rule. It is in *Greibach Normal Form* (*GNF*) if all rules are of the forms $S \to e$ and $Z \to ay$ for a in Σ and y in $(V - \Sigma - \{S\})^*$.

We shall see that these are an increasing set of restrictions on grammars with respect to c-equivalence: there are grammars not c-equivalent to any left recursion free grammar, left recursion free grammars not c-equivalent to any grammar in wGNF and grammars in wGNF not c-equivalent to any grammar in GNF. The best known transform for eliminating left recursion, replacing X-rule set $\{X \to Xv_i \mid 1 \leqslant i \leqslant m\} \cup \{X \to u_i \mid 1 \leqslant i \leqslant r\}$ (with X not leftmost in u_i) by $\{X \to u_i, X \to u_iX' \mid 1 \leqslant i \leqslant r\} \cup \{X' \to v_i, X \to v_iX' \mid 1 \leqslant i \leqslant m\}$ with X' new, does not preserve c-equivalence in general. The obvious way of transforming a left recursion free grammar to one in wGNF involves the replacement property while transforming wGNF to GNF is akin to transforming a grammar to wCNF and none of these procedures are legal in all cases.

First we shall see that for G in wGNF, $\text{CONTROL}(\mathscr{G}(G), \mathscr{L}) \subseteq \hat{\mathscr{H}}(\mathscr{L}(G) \wedge \mathscr{L})$.

THEOREM 5.5. *If G is in* wGNF *then for every* semiAFL $\mathscr{L}$,

$$\text{CONTROL}(\mathscr{G}(G), \mathscr{L}) \subseteq \hat{\mathscr{H}}(\mathscr{L}(G) \wedge \mathscr{L}).$$

Proof. Suppose G is in wGNF and $H = (V, \Sigma, P, S)$ is a reduced grammar in $\mathscr{G}(G)$ obtained from some complete interpretation $(\tau(G), \tau)$ of G. Now H may not itself be in wGNF since we can always erase a terminal from a rule without leaving $\mathscr{G}(G)$ (but not add one). However, if $Z \to y$ is in H, either $y = e$ and $Z = S$ (since Z must be derived from a nonterminal of G which cannot appear on the right-hand side of any rule and H is reduced), or $y = y_1 y_2$ and G contains a rule $Z' \to av$ such that a is a terminal, $Z \in \tau(Z')$, $y_1 \in \tau(a)$, and $y_2 \in \tau(v)$. So if c is any terminal and $\bar{\tau}(a) = \tau(a) \cup \{cy_1\}$ and $\bar{\tau}(A) = \tau(A)$ for $A \neq a$, then $\bar{\tau}(G)$ is also a complete interpretation and if we add $Z \to cy$ to H the new grammar is a subgrammar of $\bar{\tau}(G)$ and so also in $\mathscr{G}(G)$. Thus we can always add a terminal to the left of the right-hand side of any rule of H without leaving $\mathscr{G}(G)$.

Let H be a reduced grammar in $\mathscr{G}(G)$ and C a control set in $\mathscr{L}$. We simultaneously construct an e-free regular substitution σ on C and a new grammar H' in $\mathscr{G}(G)$. We can assume that the rule names are distinct from all symbols in V. If p is a rule $Z \to y$ other than $S \to e$, let H' contain $Z \to py$ with p regarded as a terminal and let $\sigma(p) = \Sigma^* p \Sigma^*$. If p is $S \to e$, let G' also contain $S \to e$ and let $\sigma(p) = \{p\}$. Let h be the homomorphism which is the identity on Σ and erases everything else. Then as discussed above, $L(H, C) = h(L(H') \cap \sigma(C)) \in \hat{\mathscr{H}}(\mathscr{L}(G) \wedge \mathscr{L})$. ∎

Remark. Theorem 5.5 holds with the weaker hypothesis that $\mathscr{L}$ is closed under e-free regular substitution.

COROLLARY 5.6. *If G is c-equivalent to a grammar in* wGNF, *then for every* semiAFL $\mathscr{L}$, $\text{CONTROL}(\mathscr{G}(G), \mathscr{L}) \subseteq \hat{\mathscr{H}}(\mathscr{L}(G) \wedge \mathscr{L})$.

Proof. The point is that for any semiAFL $\mathscr{L}$, $\hat{\mathscr{H}}(\mathscr{L}(G) \wedge \mathscr{L}) = \hat{\mathscr{H}}(\mathscr{L}(G) \wedge \hat{\mathscr{H}}(\mathscr{L}))$ [8]. So if $G \equiv^c H$ and H is in wCNF then for any semiAFL $\mathscr{L}$,

$$\begin{aligned}&\text{CONTROL}(\mathscr{G}(G), \mathscr{L})\\ &\quad \subseteq \text{CONTROL}(\mathscr{G}(G), \hat{\mathscr{H}}(\mathscr{L}))\\ &\quad = \text{CONTROL}(\mathscr{G}(H), \hat{\mathscr{H}}(\mathscr{L})) \subseteq \hat{\mathscr{H}}(\mathscr{L}(H) \wedge \hat{\mathscr{H}}(\mathscr{L})) = \hat{\mathscr{H}}(\mathscr{L}(G) \wedge \mathscr{L}). \quad \blacksquare\end{aligned}$$

A grammar in GNF is in both wCNF and WGNF, so we see that if G is in GNF then for every semiAFL $\mathscr{L}$, $\text{CONTROL}(\mathscr{G}(G), \mathscr{L}) = \hat{\mathscr{H}}(\mathscr{L}(G) \wedge \mathscr{L})$.

In some cases it suffices for G to be left recursion free.

LEMMA 5.7. *If G is left recursion free and either has no erasing rules or the only erasing rule is $S \to e$ for a start symbol S not appearing on the right-hand side of any rule, then G is c-equivalent to some grammar in* wGNF.

Proof. We can assume that G is reduced. Let G have n nonterminals and satisfy the hypotheses of this lemma. Let $W(X)$ contain all terminal strings derivable from X within n steps plus all strings derivable from X in exactly n steps of a left-to-right derivation. Since G is left recursion free and has no erasing rules except perhaps $S \to e$ which can only be applied initially, each member of $W(X)$ starts with a terminal for $X \neq S$ while each member of $W(S)$ starts with a terminal except perhaps for e. Then we can apply Corollary 2.5 repeatedly for each X and $W(X)$, and the new grammars will not violate the hypotheses of this lemma. Hence the grammar with X-rule set $\{X \to w \mid w \in W(X)\}$ for every nonterminal X is c-equivalent to G and is in wGNF. $\blacksquare$

Any ldb grammar is c-equivalent to one which is monotonic and the transforms involved do not introduce new left recursions. So we have the following result.

THEOREM 5.8. *If G is left recursion free and either is* ldb *or has no erasing rules then for every* semiAFL $\mathscr{L}$,

$$\text{CONTROL}(\mathscr{G}(G), \mathscr{L}) \subseteq \hat{\mathscr{H}}(\mathscr{L}(G) \wedge \mathscr{L}).$$

Now we can exhibit examples to show that left recursion free, wGNF and GNF are an increasing sequence of restrictions on grammars with respect to c-equivalence. The grammar G_R studied before is monotonic and linear context-free. We saw that for any full semiAFL $\mathscr{L}$, $\hat{\mathscr{H}}(\mathscr{L}(G_R) \wedge \mathscr{L}) = \mathscr{L}$ and $\text{CONTROL}(\mathscr{G}(G_R), \mathscr{L}) = \mathscr{L}^R$. We claim that G_R is not c-equivalent to any left recursion free grammar. Suppose G_R is c-equivalent to a left recursion free grammar H. Since G_R is nontrivial, H is nontrivial. By Theorem 4.4 either H is c-equivalent to some ldb subgrammar, or $\text{CONTROL}(\mathscr{G}(H), \text{LINEARL}) = \text{RE}$. In the latter case, H is not c-equivalent to G_R since $\text{CONTROL}(\mathscr{G}(G_R), \text{LINEARL}) = \text{LINEARL}$. Hence H must be c-equivalent to some ldb subgrammar H'. But H'

is also left recursion free so by Theorem 5.8, for every full semiAFL $\mathscr{L}$, $\mathscr{L}^R = \text{CONTROL}(\mathscr{G}(G_R), \mathscr{L}) = \text{CONTROL}(\mathscr{G}(H), \mathscr{L}) = \text{CONTROL}(\mathscr{G}(H'), \mathscr{L}) \subseteq \hat{\mathscr{H}}(\mathscr{L}(H') \wedge \mathscr{L}) = \hat{\mathscr{H}}(\mathscr{L}(H) \wedge \mathscr{L}) = \hat{\mathscr{H}}(\mathscr{L}(G_R) \wedge \mathscr{L}) = \mathscr{L}$ which is impossible if $\mathscr{L}$ is not closed under reversal. Thus H cannot exist.

COROLLARY 5.9. *The linear context-free grammar* $G_R = (\{S, a\}, \{a\}, \{S \to Sa, S \to a\}, S)$ *is not c-equivalent to any left recursion free grammar.*

Now consider the grammar $G = (\{S, B, a\}, \{a\}, \{S \to aSB, B \to e, S \to a\}, S)$. This grammar does not have any self-embedding symbols (note that $S \to aSB$ but B generates only the empty string) and so $\mathscr{L}(G) = \text{REGULARL}$ [5]. So for any full semiAFL $\mathscr{L}$, $\hat{\mathscr{H}}(\mathscr{L}(G) \wedge \mathscr{L}) = \mathscr{L}$. But G contains a pse symbol which is not ldb (namely, S) and so by Corollary 4.3, $\text{CONTROL}(\mathscr{G}(G), \text{LINEARL}) = \text{RE}$. Thus G is left recursion free but is not c-equivalent to any grammar in wGNF.

COROLLARY 5.10. *There is a grammar which is left recursion free and has no self-embedding symbols and is not c-equivalent to any grammar in* wGNF.

These examples show that in Theorem 5.8 the condition that G either be ldb or have no erasing rules cannot be eliminated.

The grammar G_L with production set $\{S \to aSa, S \to a\}$ is in wGNF but, as we have seen, is not c-equivalent to any grammar in wCNF and hence not to any grammar in GNF.

COROLLARY 5.11. *There is a linear context-free grammar which is in* wGNF *but is not c-equivalent to any grammar in* GNF *or in* wCNF.

The grammar $G = (\{S, A, a\}, \{a\}, \{S \to SA, S \to a, A \to a\}, S)$ is in CNF. As we saw in Section 4, since it has a pse symbol which is not ldb, $\text{CONTROL}(\mathscr{G}(G), \text{LINEARL}) = \text{RE}$. Since it is not self-embedding, $\mathscr{L}(G) = \text{REGULARL}$ [5] and so again $\hat{\mathscr{H}}(\mathscr{L}(G) \wedge \text{LINEARL}) = \text{LINEARL}$. Thus G is not c-equivalent to any grammar in wGNF.

COROLLARY 5.12. *There is a grammar which is not self-embedding and is in* CNF *but is not c-equivalent to any grammar in* wGNF.

Using Theorem 3.1 and Lemma 5.1 of Ref. [5] and induction on the number of nonterminals, the following can be established.

LEMMA 5.13. *Every grammar is g-equivalent to one in* GNF.

Now we can put together the preceding lemmas and theorems and give a necessary and sufficient condition for (**) to hold.

THEOREM 5.14. *A grammar G is c-equivalent to some grammar in* GNF *if and only if*

$$\text{CONTROL}(\mathscr{G}(G), \mathscr{L}) = \hat{\mathscr{H}}(\mathscr{L}(G) \wedge \mathscr{L})$$

for every full semiAFL $\mathscr{L}$.

Proof. If G is c-equivalent to a grammar H in GNF then H is in both wGNF and wCNF and so (**) holds for H. Since G is also g-equivalent to H we have for every full semiAFL $\mathscr{L}$

$$\text{CONTROL}(\mathscr{G}(G), \mathscr{L}) = \text{CONTROL}(\mathscr{G}(H), \mathscr{L}) = \hat{\mathscr{H}}(\mathscr{L}(H) \wedge \mathscr{L}) = \hat{\mathscr{H}}(\mathscr{L}(G) \wedge \mathscr{L}).$$

On the other hand, any grammar G is g-equivalent to some grammar H in GNF. If (**) holds for G then for every full semiAFL $\mathscr{L}$

$$\text{CONTROL}(\mathscr{L}(G), \mathscr{L}) = \hat{\mathscr{H}}(\mathscr{L}(G) \wedge \mathscr{L}) = \hat{\mathscr{H}}(\mathscr{L}(H) \wedge \mathscr{L}) = \text{CONTROL}(\mathscr{G}(H), \mathscr{L})$$

so G is also c-equivalent to H. ∎

We can also see that (**) holds whenever $\mathscr{L}(G)$ is the whole family of context-free languages. We use the following fact about self-expansive symbols established in Ref. [5].

DEFINITION. A nonterminal X in a grammar G is *self-expansive* if there are strings u, v, w such that $X \overset{+}{\Rightarrow} uXvXw$. A grammar is *nonexpansive* if none of its nonterminals are expansive.

LEMMA 5.15. (Cremers and Ginsburg). *For any reduced grammar G, $\mathscr{L}(G) = \text{CFL}$ if and only if G contains a nontrivial self-expansive symbol.*

THEOREM 5.16. *If $\mathscr{L}(G) = \text{CFL}$, then for any* semiAFL $\mathscr{L}$,

$$\text{CONTROL}(\mathscr{G}(G), \mathscr{L}) = \hat{\mathscr{H}}(\text{CFL} \wedge \mathscr{L}).$$

Proof. The idea is to construct a grammar H in $\mathscr{G}(G)$ which is in wCNF and is g-equivalent to G. Since G is itself a context-free grammar, (*) yields for every semiAFL $\mathscr{L}$,

$$\text{CONTROL}(\mathscr{G}(G), \mathscr{L}) \subseteq \text{CONTROL}(\text{CFG}, \mathscr{L}) = \hat{\mathscr{H}}(\text{CFL} \wedge \mathscr{L}) = \hat{\mathscr{H}}(\mathscr{L}(G) \wedge \mathscr{L}).$$

On the other hand, the existence of H will yield together with Theorem 5.2

$$\hat{\mathscr{H}}(\mathscr{L}(G) \wedge \mathscr{L}) = \hat{\mathscr{H}}(\mathscr{L}(H) \wedge \mathscr{L}) \subseteq \text{CONTROL}(\mathscr{G}(H), \mathscr{L}) \subseteq \text{CONTROL}(\mathscr{G}(G), \mathscr{L})$$

for every semiAFL $\mathscr{L}$ and so

$$\text{CONTROL}(\mathscr{G}(G), \mathscr{L}) = \hat{\mathscr{H}}(\text{CFL} \wedge \mathscr{L}).$$

So it remains to construct H. The existence of H follows almost directly from Lemma 5.15 cited above. We can assume G to be reduced. Let $G = (V, \Sigma, P, S)$ and let h be the homomorphism which is the identity on $V - \Sigma$ but erases all of Σ. Let

$$P' = \{Z \to Ah(y) \mid Z \to Ay \in P, A \in V\} \cup \{Z \to e \mid Z \to e \in P\}$$

and $H = (V, \Sigma, P', S)$. Clearly H is in $\mathscr{G}(G)$ and is in wCNF. Since $\mathscr{L}(G) = \text{CFL}$, by Lemma 5.15 G contains a nontrivial self-expansive nonterminal, say X. Clearly X

is also self-expansive in H. Because X is nontrivial in G, X must generate some nonempty string and as P' does not introduce any new erasing rules (though it may decrease the number of terminals in a production), X must also generate some nonempty terminal string in H. Since H is reduced and X is self-expansive, X must generate infinitely many terminal strings in H. Thus $\mathscr{L}(H) = \text{CFL} = \mathscr{L}(G)$, so H and G are g-equivalent. ∎

COROLLARY 5.17. *If $\mathscr{L}(G) = \text{CFL}$, every grammar g-equivalent to G is c-equivalent to G.*

There are still some open questions regarding the use of (**) to characterize the G-control operator for G not left derivation bounded. If G is reduced and not self-embedding then $\mathscr{L}(G) = \text{REGULARL}$ so as we have seen (**) holds if and only if G is c-equivalent to a regular grammar. Thus if G is reduced and not self-embedding and not c-equivalent to any left derivation bounded grammar, (**) cannot hold. However, if G is self-embedding and not c-equivalent to any left derivation bounded grammar then

$$\text{RE} = \hat{\mathscr{H}}(\text{LINEARL} \wedge \text{LINEARL}) \subseteq \hat{\mathscr{H}}(\mathscr{L}(G) \wedge \text{LINEARL}) \subseteq \text{RE}$$

so by Theorem 4.4 we have the following characterization (***)

(***) For every semiAFL $\mathscr{L}$ such that $\text{LINEARL} \subseteq \mathscr{L} \subseteq \text{RE}$,

$$\text{CONTROL}(\mathscr{G}(G), \mathscr{L}) = \hat{\mathscr{H}}(\mathscr{L}(G) \wedge \mathscr{L}).$$

This is not quite (**) since of course it excludes many full semiAFLs. It is an open question whether (**) holds whenever G is self-embedding and not c-equivalent to any left derivation bounded grammar. Again, property (***) makes counterexamples hard to find.

6. LINEAR GRAMMARS, HOMOMORPHIC REPLICATION, AND HIERARCHY THEOREMS

We have just investigated one type of characterization for the G-control operator acting on a full semiAFL $\mathscr{L}$, a characterization expressed in terms of homomorphisms, intersections, $\mathscr{L}(G)$, and $\mathscr{L}$. We saw that it holds if and only if G is c-equivalent to a grammar in Greibach Normal Form and that it never holds for G left derivation bounded but essentially nonregular (i.e., not c-equivalent to any regular grammar). Now we investigate possible characterizations for G left derivation bounded and find that we use operators of a different kind, namely, several types of homomorphic replication operators. (The operation of homomorphic replication was introduced by Ginsburg and Spanier in Ref. [10].)

First we shall examine the situation for linear grammars, and then in Section 7 extend our results to nonterminal bounded grammars. In these cases not only do we get elegant characterizations but they lead to very powerful hierarchy theorems. We shall see that if G is nonterminal bounded and self-embedding then the G-control operator is *hierarchial* in the following sense. If a full semiAFL $\mathscr{L}$ is not closed under the G-control operator

then for every $k \geqslant 1$ the k-fold iteration of the G-control operator on $\mathscr{L}$ is properly contained in the $(k+1)$-fold iteration of the G-control operator on $\mathscr{L}$.

We start with a characterization theorem for CONTROL($\mathscr{G}(G), \mathscr{L}$) when G is metalinear and $\mathscr{L}$ is a full semiAFL.

DEFINITION. A grammar $G = (V, \Sigma, P, S)$ is *metalinear of degree n* if all the rules of P are of the forms $S \to X_1 \cdots X_k$, $X \to \alpha Y \beta$, $X \to \gamma$ for $k \leqslant n$, $X_1, \ldots, X_k$, X, $Y \in V - \Sigma - \{S\}$ and α, β, γ in Σ^*. Let METAnG be the family of metalinear grammars of degree n and let METAnL be the corresponding family of languages.

It is readily seen that LINEARG $= \mathscr{G}(G)$ for the context-free grammar $G = (\{S, a\}, \{a\}, \{S \to aSa, S \to a\}, S)$. Since G is ldb, CONTROL(LINEARG, $\mathscr{L}$) is a full semiAFL for any semiAFL $\mathscr{L}$. Similarly, METAnG $= \mathscr{G}(G_n)$ for the grammar $G_n = (\{S, a\}, \{a\}, \{S \to X^k \mid 1 \leqslant k \leqslant n\} \cup \{X \to aXa, X \to a\}, S)$ and so CONTROL (METAnG, $\mathscr{L}$) is a full semiAFL for any semiAFL $\mathscr{L}$. We obtain more detailed information through the use of certain types of homomorphic replication operators.

DEFINITION. Let $n \geqslant 1$. Let $L \subseteq (\Sigma \cup \{c\})^*$ for a finite vocabulary Σ and a symbol c not in Σ. For homomorphisms h_1 and h_2, let

$$f_{n,c}(h_1, h_2, L) = \{h_1(w_1)\, h_2(w_1^R) \cdots h_1(w_n)\, h_2(w_n^R) \mid w_1 c \cdots w_n c \in L\}.$$

For a family of languages $\mathscr{L}$, let

$$n\text{-HOM}(\mathscr{L}) = \{f_{n,c}(h_1, h_2, L) \mid L \in \mathscr{L},\ h_1, h_2 \text{ homomorphisms}\}.$$

For the linear context-free case we can use somewhat simpler operations.

DEFINITION. For a language L and homomorphisms h_1 and h_2, let

$$\rho(L) = \{ww^R \mid w \in L\},$$
$$f(h_1, h_2, L) = \{h_1(w)\, h_2(w^R) \mid w \in L\}.$$

For a family of languages $\mathscr{L}$, let

$$\text{HOM}(\mathscr{L}) = \{f(h_1, h_2, L) \mid L \in \mathscr{L},\ h_1, h_2 \text{ homomorphisms}\}.$$

Our main characterization theorem says that for any semiAFL $\mathscr{L}$, CONTROL (LINEARG, $\mathscr{L}$) coincides with HOM($\mathscr{L}$) and for any $n \geqslant 1$, CONTROL(METAnG, $\mathscr{L}$) coincides with n-HOM($\mathscr{L}$).

THEOREM 6.1. *Let $\mathscr{L}$ be a* semiAFL. *For each $n \geqslant 1$,*

$$\text{CONTROL(METAnG, } \mathscr{L}) = n\text{-HOM}(\mathscr{L}).$$

For $n = 1$

$$\text{CONTROL(LINEARG, } \mathscr{L}) = \text{HOM}(\mathscr{L}) = 1\text{-HOM}(\mathscr{L}) = \hat{\mathscr{M}}(\{\rho(L) \mid L \in \mathscr{L}\}).$$

Proof. First consider L in $\mathscr{L}$ and homomorphisms h_1 and h_2. Let $L \subseteq (\Sigma \cup \{c\})^*$ for a finite vocabulary Σ and a symbol c not in Σ. Let $\mathscr{S}$ be a new symbol. Let H_n be the grammar with initial symbol S and productions $S \to X^n$ labeled $\mathscr{S}$, $X \to e$ labeled c, and for each a in Σ, $X \to h_1(a)\, X h_2(a)$ labeled a. Then H_n is metalinear of degree n, $\mathscr{S}L$ is in $\mathscr{L}$, and $f_{n,c}(h_1, h_2, L) = L(H_n, \mathscr{S}L) \in$ CONTROL(METAnG, $\mathscr{L}$). So n-HOM($\mathscr{L}$) $\subseteq$ CONTROL(METAnG, $\mathscr{L}$).

Let G_n be the grammar with initial symbol S and production set $\{S \to X^k \mid 1 \leqslant k \leqslant n\} \cup \{X \to aXa,\ X \to a\}$. Proposition 2.13 tells us that to show CONTROL(METAnG, $\mathscr{L}$) $\subseteq$ n-HOM($\mathscr{L}$) it suffices to show $L(\sigma(G_n), C)$ in n-HOM($\mathscr{L}$) for each C in $\mathscr{L}$ and each simple interpretation σ of G_n. Let σ be a simple interpretation of G_n. Let c be a new symbol. Let $\mathscr{S}_k$ label the production $S \to X^k$ for $1 \leqslant k \leqslant n$. Let P_1 be the set of labels of productions of $\sigma(G_n)$ of the form $X \to \alpha X \beta$ and P_2 the set of labels of productions of $\sigma(G_n)$ of the form $X \to \gamma$, α, β, and γ being terminal strings. We define homomorphisms g, h_1, and h_2 on production names as follows. For each k, $g(\mathscr{S}_k) = h_1(\mathscr{S}_k) = h_2(\mathscr{S}_k) = e$. If p is a production $X \to \alpha X \beta$ in P_1, then $g(p) = p$, $h_1(p) = \alpha$, and $h_2(p) = \beta$. If p is a production $X \to \gamma$ in P_2, then $g(p) = pc$, $h_1(p) = \gamma$, and $h_2(p) = e$. Finally, let $L = \bigcup_{1 \leqslant k \leqslant n} g(C \cap \mathscr{S}_k (P_1{}^* P_2)^k) c^{n-k}$. Then L is in $\mathscr{L}$ since $\mathscr{L}$ is a semiAFL and $L(\sigma(G_n), C) = f_{n,c}(h_1, h_2, L) \in n$-HOM($\mathscr{L}$). Hence CONTROL(METAnG, $\mathscr{L}$) $\subseteq$ n-HOM($\mathscr{L}$) so CONTROL(METAnG, $\mathscr{L}$) $= n$-HOM($\mathscr{L}$).

For $n = 1$, it should be clear that HOM($\mathscr{L}$) $=$ 1-HOM($\mathscr{L}$) $=$ CONTROL (METAlG, $\mathscr{L}$) $=$ CONTROL(LINEARG, $\mathscr{L}$). If L is in $\mathscr{L}$, obviously $\rho(L)$ is in HOM($\mathscr{L}$). Since CONTROL(LINEARG, $\mathscr{L}$) $=$ HOM($\mathscr{L}$) is a full semiAFL, $\mathscr{M}(\{\rho(L) \mid L \in \mathscr{L}\}) \subseteq$ HOM($\mathscr{L}$). On the other hand, if h_1 and h_2 are homomorphisms, L is in $\mathscr{L}$ and c is a new symbol, one can readily construct an a-transducer M to take a word $wccw^R$ into $h_1(w)\, h_2(w^R)$ and thus $f(h_1, h_2, L) = M(\rho(Lc))$. Hence HOM($\mathscr{L}$) $\subseteq \mathscr{M}(\{\rho(L) \mid L \in \mathscr{L}\})$ and we are done. ∎

The operation $f(h_1, h_2, L)$ on L is a special example of a homomorphic replication [10, 17]. We state without proof the following lemmas which can be obtained from Lemma 5.7 and Corollary 1 of p. 71 of Ref. [17]. They give consequences of particular sets being in HOM($\mathscr{L}$).

LEMMA 6.2. *Let $L \subseteq \Sigma^+$, $c \notin \Sigma$. If $\mathscr{L}$ is a full* semiAFL *such that $\rho(Lc)$ is in* HOM($\mathscr{L}$), *then L is in $\mathscr{L}$.*

LEMMA 6.3. *Let $L_1, L_2 \subseteq \Sigma^+$, $c \notin \Sigma$. If $\mathscr{L}$ is a full* semiAFL *such that $L_1 c L_2$ is in* HOM($\mathscr{L}$) *then either L_1 is in $\mathscr{L}$ or L_2 is in $\mathscr{L}^R$.*

LEMMA 6.4. *Let $L \subseteq \Sigma^+$, $c \notin \Sigma$. If $\mathscr{L}$ is a full* semiAFL *such that $\rho(Lc)$ is in $\mathscr{F}(\mathscr{L})$, then $\rho(Lc)$ and L are both in $\mathscr{L}$.*

These properties are close to the properties of a "syntactic operator" in Ref. [17] and in some respects are even stronger. So they can be used to yield hierarchy theorems without referring to the particular full semiAFL under discussion. For convenience let us use the following abbreviations.

DEFINITION. For any family $\mathscr{G}$ of grammars and any family $\mathscr{L}$ of languages, let $\text{CONTROL}(\mathscr{G}, \mathscr{L}) = \mathscr{L}$ and for $k \geqslant 0$, let $\text{CONTROL}_{k+1}(\mathscr{G}, \mathscr{L}) = \text{CONTROL}(\mathscr{G}, \text{CONTROL}_k(\mathscr{G}, \mathscr{L}))$. Let $\text{CONTROL}_\infty(\mathscr{G}, \mathscr{L}) = \bigcup_{k \geqslant 0} \text{CONTROL}_k(\mathscr{G}, \mathscr{L})$. Define $\text{HOM}_k(\mathscr{L})$, $\text{HOM}_\infty(\mathscr{L})$, $n\text{-HOM}_k(\mathscr{L})$, and $n\text{-HOM}_\infty(\mathscr{L})$ similarly.

Thus $\text{CONTROL}_k(\mathscr{G}, \mathscr{L})$ is the k-fold iteration of the $\mathscr{G}$-control operator on $\mathscr{L}$. $\text{CONTROL}_\infty(\mathscr{G}, \mathscr{L})$ is the closure of $\mathscr{L}$ under the $\mathscr{G}$-control operator.

We are interested in several special cases. First we consider $\mathscr{G} = \text{LINEARG}$ and the linear-control operator which we have just seen to be equivalent to the homomorphic replication operator $\text{HOM}(\mathscr{L})$. Theorem 6.1 and Lemma 6.2 together tell us that the process of placing controls on linear context-free languages preserves not only inclusion but also "noninclusion" among full semiAFLs. Thus if $\mathscr{L}_1$ is not included in $\mathscr{L}_2$, then $\text{CONTROL}(\text{LINEARG}, \mathscr{L}_1)$ is not included in $\text{CONTROL}(\text{LINEARG}, \mathscr{L}_2)$. Putting this in the positive direction we have Theorem 6.5 below.

THEOREM 6.5. *For full* semiAFLs $\mathscr{L}_1$ *and* $\mathscr{L}_2$,

(a) $\text{CONTROL}(\text{LINEARG}, \mathscr{L}_1) \subseteq \text{CONTROL}(\text{LINEARG}, \mathscr{L}_2)$ if and only if $\mathscr{L}_1 \subseteq \mathscr{L}_2$,

and

(b) $\text{CONTROL}(\text{LINEARG}, \mathscr{L}_1) = \text{CONTROL}(\text{LINEARG}, \mathscr{L}_2)$ if and only if $\mathscr{L}_1 = \mathscr{L}_2$.

Proof. Obviously (b) follows from two applications of (a). By definition, if $\mathscr{L}_1 \subseteq \mathscr{L}_2$ then $\text{CONTROL}(\text{LINEARG}, \mathscr{L}_1)$ is included in $\text{CONTROL}(\text{LINEARG}, \mathscr{L}_2)$. Now suppose that $\text{CONTROL}(\text{LINEARG}, \mathscr{L}_1) \subseteq \text{CONTROL}(\text{LINEARG}, \mathscr{L}_2)$. Let L be any member of $\mathscr{L}_1$ and let c be any symbol not appearing in any word of L. Since $\mathscr{L}_1$ and $\mathscr{L}_2$ as full semiAFLs are closed under removal and addition of the empty word, we can assume without loss of generality that L does not contain the empty word. Now Lc is also in $\mathscr{L}_1$ and by Theorem 6.1, $\rho(Lc)$ is in $\text{CONTROL}(\text{LINEARG}, \mathscr{L}_1)$ and so in $\text{CONTROL}(\text{LINEARG}, \mathscr{L}_2)$. Thus by Lemma 6.2, L is in $\mathscr{L}_2$. Hence $\mathscr{L}_1$ is contained in $\mathscr{L}_2$. ∎

Among the various possible corollaries of Theorem 6.5, we can mention the following.

COROLLARY 6.6. *For full* semiAFLs $\mathscr{L}_1$ *and* $\mathscr{L}_2$ *and* $k \geqslant 0$

$\text{CONTROL}_k(\text{LINEARG}, \mathscr{L}_1) \subseteq \text{CONTROL}_k(\text{LINEARG}, \mathscr{L}_2)$ *if and only if* $\mathscr{L}_1 \subseteq \mathscr{L}_2$,
$\text{CONTROL}_k(\text{LINEARG}, \mathscr{L}_1) = \text{CONTROL}_k(\text{LINEARG}, \mathscr{L}_2)$ *if and only if* $\mathscr{L}_1 = \mathscr{L}_2$.

COROLLARY 6.7. *For a full* semiAFL $\mathscr{L}$ *if* $\mathscr{L} \neq \text{CONTROL}(\text{LINEARG}, \mathscr{L})$ *(i.e., if* $\mathscr{L}$ *is not closed under using members of* $\mathscr{L}$ *to control linear context-free grammars), then for all* $k \geqslant 0$

$$\text{CONTROL}_k(\text{LINEARG}, \mathscr{L}) \neq \text{CONTROL}_{k+1}(\text{LINEARG}, \mathscr{L}).$$

COROLLARY 6.8. *If $\mathscr{L}_1$ and $\mathscr{L}_2$ are incomparable full* semiAFLs, *then for all $k \geqslant 0$,* $\text{CONTROL}_k(\text{LINEARG}, \mathscr{L}_1)$ *and* $\text{CONTROL}_k(\text{LINEARG}, \mathscr{L}_2)$ *are incomparable full* semiAFLs.

From Lemma 6.3 we can establish at once the following lemma.

LEMMA 6.9. *Let $\mathscr{L}$ be a full* semiAFL. *Let $L \subseteq \Sigma^+$ and $c \notin \Sigma$. If L and L^R are not in $\mathscr{L}$, then for all $k \geqslant 0$, $(Lc)^{2^k}$ is not in* $\text{CONTROL}_k(\text{LINEARG}, \mathscr{L})$ *and $(Lc)^+$ is not in* $\text{CONTROL}_\infty(\text{LINEARG}, \mathscr{L})$.

Proof. We proceed to show by induction on k that $(Lc)^{2^k}$ is not in CONTROL_k (LINEARG, $\mathscr{L}$). For $k = 0$ the result is immediate from the hypothesis and the fact that Lc is in $\mathscr{L}$ if and only if L is in $\mathscr{L}$ since $\mathscr{L}$ is a full semiAFL. Suppose we have established the result for $k \geqslant 0$. Let d be a new symbol and let $L' = (Lc)^{2^k}$. Now $L'L' = (Lc)^{2^{k+1}}$ and $L'dL'$ are a-tranducer mappings of each other and so either both or none are in $\text{CONTROL}_{k+1}(\text{LINEARG}, \mathscr{L})$. So by Lemma 6.3, if $L'L'$ is in CONTROL_{k+1} (LINEARG, $\mathscr{L}$) either L' is in $\text{CONTROL}_k(\text{LINEARG}, \mathscr{L})$ or it is in $(\text{CONTROL}_k$ (LINEARG, $\mathscr{L}))^R$. The first case contradicts the induction hypothesis. Since we always have $(f(h_1, h_2, A))^R = f(h_2{}^R, h_1{}^R, A)$ (where h^R is the homomorphism defined by $h^R(a) = (h(a))^R$ for every symbol a), $\text{CONTROL}_k(\text{LINEARG}, \mathscr{L})$ is always closed under reversal for $k \geqslant 1$ yielding L' in $\text{CONTROL}_k(\text{LINEARG}, \mathscr{L})$ a contradiction. For $k = 0$, we observe that $L' = Lc$ and by hypothesis L^R is not in $\mathscr{L}$ and so L and L' are not in $\mathscr{L}^R = (\text{CONTROL}_0)\text{LINEARG}, \mathscr{L})^R$. Now if $(Lc)^+$ is in CONTROL_∞ (LINEARG, $\mathscr{L}$), it is in some particular full semiAFL $\text{CONTROL}_k(\text{LINEARG}, \mathscr{L})$. Thus $(Lc)^+ \cap (\Sigma^+c)^{2^k} = (Lc)^{2^k}$ is in $\text{CONTROL}_k(\text{LINEARG}, \mathscr{L})$, a contradiction. ∎

We shall use Lemma 6.9 to show that if a full semiAFL is not closed under concatenation, then applications of the linear control operator will not change the situation. Similarly, if $\mathscr{L}$ is not closed under Kleene+, neither is $\text{CONTROL}_\infty(\text{LINEARG}, \mathscr{L})$. These results hold in a very strong sense: if $\mathscr{L}$ is not closed under concatenation, its concatenation closure will not be in any $\text{CONTROL}_k(\text{LINEARG}, \mathscr{L})$ (but may be in $\text{CONTROL}_\infty(\text{LINEARG}, \mathscr{L})$ as is the case when $\mathscr{L}$ is the family of linear context-free languages) and if $\mathscr{L}$ is not closed under Kleene+ then its closure under Kleene+ (and so $\mathscr{F}(\mathscr{L})$) is not in $\text{CONTROL}_\infty(\text{LINEARG}, \mathscr{L})$.

THEOREM 6.10. *Let $\mathscr{L}$ be a full* semiAFL.

(a) *If $\mathscr{L}$ is not closed under Kleene + (i.e., is not a full* AFL), *then* CONTROL_∞ (LINEARG, $\mathscr{L}$) *does not contain the closure of $\mathscr{L}$ under Kleene+ and hence is not closed under Kleene+.*

(b) *If $\mathscr{L}$ is not closed under concatenation, then for all $k \geqslant 0$,* CONTROL_k (LINEARG, $\mathscr{L}$) *does not contain the closure of $\mathscr{L}$ under concatenation and hence is not closed under concatenation and* $\text{CONTROL}_\infty(\text{LINEARG}, \mathscr{L})$ *does not contain the closure of $\mathscr{L}$ under Kleene+ and hence is not closed under Kleene+.*

Proof. In view of Lemma 6.9, to establish part (a) we need only show that if $\mathscr{L}$ is not closed under Kleene+ there is a language L in $\mathscr{L}$ such that L^+ is in neither $\mathscr{L}$ nor $\mathscr{L}^R$.

If $\mathscr{L} = \mathscr{L}^R$, the existence of L is immediate from the hypothesis. Otherwise, let L_1 be a language in $\mathscr{L}$ such that L_1^+ is not in $\mathscr{L}$ and L_2 a language in $\mathscr{L} - \mathscr{L}^R$. Let L_1 and L_2 be contained in Σ^* for a finite vocabulary Σ and let d_1 and d_2 be new symbols. Finally let $L = d_1L_1 \cup d_2L_2$. Thus L is in $\mathscr{L}$. If L^+ were in $\mathscr{L}$, then $\mathscr{L}$ would contain $(d_1L_1)^+ = L^+ \cap (d_1\Sigma^*)^+$ and hence L_1^+, a contradiction. If L^+ were in $\mathscr{L}^R$, then $\mathscr{L}^R$ would contain $d_2L_2 = L^+ \cap d_2\Sigma^*$ and hence L_2, another contradiction. Thus in all cases, L^+ is in neither $\mathscr{L}$ nor $\mathscr{L}^R$ although L is in $\mathscr{L}$.

We now wish to show that if $\mathscr{L}$ is not closed under concatenation then there is a language L which is the concatenation of two members of $\mathscr{L}$ but is in neither $\mathscr{L}$ nor $\mathscr{L}^R$. Then part (b) follows at once from Lemma 6.9, part (*a*) of this theorem and the fact that a semiAFL not closed under concatenation is a fortiori not closed under Kleene+ [19].

Again we need only consider the two cases, $\mathscr{L} = \mathscr{L}^R$ and $\mathscr{L} \neq \mathscr{L}^R$. If $\mathscr{L} = \mathscr{L}^R$, then the existence of L follows from the nonclosure of $\mathscr{L}$ under concatenation. Otherwise, let L_1 and L_2 be two members of $\mathscr{L}$ such that L_1L_2 is not in $\mathscr{L}$ and let L_3 be in $\mathscr{L} - \mathscr{L}^R$; assume all are contained in Σ^* for some finite vocabulary Σ. Then let $L = d_1L_1 \cup d_2L_2 \cup d_3L_3$ for new symbols d_1, d_2, and d_3. Clearly L is in Σ. If LL were in $\mathscr{L}$, then $d_1L_1d_2L_2 = LL \cap d_1\Sigma^*d_2\Sigma^*$ and hence L_1L_2 would also be in $\mathscr{L}$. If LL were in $\mathscr{L}^R$, then $d_3L_3d_3L_3 = LL \cap d_3\Sigma^*d_3\Sigma^*$ would be in $\mathscr{L}^R$ but L_3 can be obtained from $d_3L_3d_3L_3$ by an *a*-transducer mapping. Thus LL is neither in $\mathscr{L}$ nor in $\mathscr{L}^R$, although L is in $\mathscr{L}$. ∎

THEOREM 6.11. *If $\mathscr{L}$ is a full* semiAFL *such that $\mathscr{L} \neq$* CONTROL(LINEARG, $\mathscr{L}$), *then for each $k \geqslant 1$,* CONTROL$_k$(LINEARG, $\mathscr{L}$) *is not closed under concatenation and* CONTROL$_\infty$(LINEARG, $\mathscr{L}$) *is not closed under Kleene+.*

Proof. In view of Theorem 6.10, we need only show that CONTROL(LINEARG, $\mathscr{L}$) is not closed under concatenation if $\mathscr{L} \neq$ CONTROL(LINEARG, $\mathscr{L}$). Now by Theorem 6.1, if $\mathscr{L} \neq$ CONTROL(LINEARG, $\mathscr{L}$), there is a language L in $\mathscr{L}$ such that $\rho(L)$ is not in $\mathscr{L}$; we can assume that L is *e*-free. Since $\rho(L) = (\rho(L))^R$, $\rho(L)$ is also not in $\mathscr{L}^R$. Hence for any new symbol c, $\rho(L)\,c\rho(L)c$ is not CONTROL(LINEARG, $\mathscr{L}$) by Lemma 6.9, although $\rho(L)c$ is in CONTROL(LINEARG, $\mathscr{L}$) by Theorem 6.1. ∎

We have now shown that if $\mathscr{L}$ is not a full AFL, we cannot obtain $\mathscr{F}(\mathscr{L})$ by application of the linear-control operator (homomorphic replication). We now use Lemma 6.4 to show the complimentary situation: if $\mathscr{L}$ is not closed under the linear-control operator, going to $\mathscr{F}(\mathscr{L})$ will not change the situation.

LEMMA 6.12. *For full* semiAFLs $\mathscr{L}_1$ *and* $\mathscr{L}_2$, CONTROL(LINEARG, $\mathscr{L}_1$) $\subseteq \mathscr{F}(\mathscr{L}_2)$ *if and only if* CONTROL(LINEARG, $\mathscr{L}_1$) $\subseteq \mathscr{L}_2$.

Proof. The "if" part is obvious. Suppose CONTROL(LINEARG, $\mathscr{L}_1$) $\subseteq \mathscr{F}(\mathscr{L}_2)$. Consider any language L in $\mathscr{L}_1$; we can assume that L is *e*-free. Let c be a new symbol. By Theorem 6.1, $\rho(Lc)$ is in $\mathscr{F}(\mathscr{L}_2)$. Then by Lemma 6.4, $\rho(Lc)$ is in $\mathscr{L}_2$. Clearly $\rho(L)$ can be obtained as an *a*-transducer mapping of $\rho(Lc)$ and hence is in $\mathscr{L}_2$. Thus $\mathscr{L}_2$ contains all languages of the form $\rho(L)$ for L in $\mathscr{L}_1$ and hence by Theorem 6.1, $\mathscr{L}_2$ contains all of CONTROL(LINEARG, $\mathscr{L}_1$). ∎

Now we can put the previous lemmas and theorems together and deduce various consequences of $\mathscr{L} \neq$ CONTROL(LINEARG, $\mathscr{L}$).

THEOREM 6.13. *Let $\mathscr{L}$ be a full* semiAFL *such that $\mathscr{L} \neq$* CONTROL(LINEARG, $\mathscr{L}$).

(1) *For all $k \geqslant 0$*

(a) $\text{CONTROL}_{k+1}(\text{LINEARG}, \mathscr{L}) - \mathscr{F}(\text{CONTROL}_k(\text{LINEARG}, \mathscr{L})) \neq \phi$, *and*

(b) $\text{CONTROL}_k(\text{LINEARG}, \mathscr{L})$ *does not contain the closure under concatenation of* CONTROL(LINEARG, $\mathscr{L}$).

(2) *For all $k \geqslant 1$*

$$\mathscr{F}(\text{CONTROL}_k(\text{LINEARG}, \mathscr{L})) - \text{CONTROL}_{k+1}(\text{LINEARG}, \mathscr{L}) \neq \phi.$$

(3) $\mathscr{F}(\text{CONTROL}(\text{LINEARG}, \mathscr{L}))$ *and* $\text{CONTROL}_\infty(\text{LINEARG}, \mathscr{L})$ *are incomparable.*

(4) *If $\mathscr{L} \neq \mathscr{F}(\mathscr{L})$ then $\mathscr{F}(\mathscr{L})$ and* $\text{CONTROL}_\infty(\text{LINEARG}, \mathscr{L})$ *are incomparable.*

Proof. Corollary 6.7 tells us the

$$\text{CONTROL}_{k+1}(\text{LINEARG}, \mathscr{L}) - \text{CONTROL}_k(\text{LINEARG}, \mathscr{L}) \neq \phi$$

and using Lemma 6.12 we see that this inequality holds if we replace CONTROL_k (LINEARG, $\mathscr{L}$) by $\mathscr{F}(\text{CONTROL}_k(\text{LINEARG}, \mathscr{L}))$. This establishes (1a).

For $k = 0$, statement (1b) is trivial. Otherwise it follows from Theorem 6.10 and the fact that CONTROL(LINEARG, $\mathscr{L}$) is not closed under concatenation by Theorem 6.11. This establishes (1b). Statement (2) is an immediate consequence of (1b), while (3) follows from (1a) and Theorems 6.10 and 6.11.

From (1a) we know that $\text{CONTROL}_1(\text{LINEARG}, \mathscr{L})$ contains a language not in $\mathscr{F}(\mathscr{L})$. Theorems 6.10 and 6.11 tell us that, on the other hand, $\mathscr{F}(\mathscr{L})$ contains a language not in any $\text{CONTROL}_k(\text{LINEARG}, \mathscr{L})$. This establishes (4). ∎

Examples

Consider the family of context-free languages, CFL. Obviously CFL $\neq$ CONTROL (LINEARG, CFL) (e.g., for $L = \{a^n b^n \mid n \geqslant 1\}$ which is even in LINEARL, $\rho(L)$ is not context-free). Hence Corollary 6.1 tells us at once that CONTROL_k(LINEARG, CFL) is properly contained in CONTROL_{k+1}(LINEARG, CFL) and for $k \geqslant 1$ none of these families is closed under concatenation. Since LINEARG $\subseteq$ LDBG, CONTROL_∞ (LINEARG, CFL) is properly contained in the family of context-sensitive languages by Theorem 2.11.

Thus Theorem 6.13 yields at once the various hierarchy results in Ref. [22]. The hierarchies in Ref. [21] can be obtained by taking as $\mathscr{L}$ the metalinear languages of degree r, METArL, as follows.

The example showing CFL $\neq$ CONTROL(LINEARG, CFL) uses a linear context-free language so we have METArL $\neq$ CONTROL(LINEARG, METArL) for $r \geqslant 1$. It is not difficult to verify using Theorem 6.1 that

$$\text{META}(2r)L \subseteq \text{CONTROL}(\text{LINEARG, METArL})$$

for $r \geqslant 1$. Hence

$$\begin{aligned}\text{CONTROL}_\infty(\text{LINEARG, REGULARL}) &= \text{CONTROL}_\infty(\text{LINEARG, LINEARL}) \\ &= \bigcup_{r \geqslant 0} \text{CONTROL}_\infty(\text{LINEARG, METArL}).\end{aligned}$$

Clearly CONTROL_∞(LINEARG, REGULARL) contains noncontext-free languages while by Theorem 6.13(4), CONTROL_∞(LINEARG, REGULARL), the closure of the regular languages under homomorphic replication, does not even contain $\mathscr{F}$(LINEARL). Hence CFL and CONTROL_∞(LINEARG, REGULARL) are incomparable. It is shown in [33] that CONTROL_∞(LINEARG, REGULARL) is the family of finite reversal checking automaton languages when finite-reversal checking automaton is defined by combining the notions of a finite-turn pushdown automaton [13] or reversal-bounded automaton [1] with that of a checking automaton [14] rather than the more restrictive definition of a finite-turn checking automaton used by Siromoney [27].

Suppose CONTROL_n(LINEARG, METArL) = CONTROL_k(LINEARG, METAsL) and, say, $n \geqslant k$. By Theorem 6.5, METAsL = CONTROL_{n-k}(LINEARG, METArL). Now the right-hand side contains noncontext-free languages for $n - k \geqslant 1$ and $r \geqslant 1$. If $n - k = 0$ we have METAsL = METArL and hence $r = s$ [18]. Thus, for $n \geqslant k$ and $r, s \geqslant 1$, CONTROL_n(LINEARG, METArL) = CONTROL_k(LINEARG METArL) if and only if $n = k$ and $r = s$ so we have a two-way hierarchy incomparable with the family of context-free languages (or even with the least AFL containing the linear context-free languages).

We can get similar results by taking as our start family $\mathscr{L}$ the family of ultralinear languages of a particular degree (cf. [13]) or the families of one counter or of nested counter languages (cf. [15]). Details are left to the reader.

We can obtain similar results for control sets on metalinear grammars by following the characterization of Theorem 6.1 by "syntactic lemmas" akin to Lemmas 6.2, 6.3, and 6.4 and finally hierarchy results like Theorems 6.10, 6.11, and 6.13. However, it is easier to obtain our hierarchy results by using the previous theorems and establishing a connection between controls on metalinear grammars and the iteration of the linear control operator. The main result is that for any semiAFL $\mathscr{L}$, and any $n \geqslant 1$, CONTROL_∞(LINEARG, $\mathscr{L}$) = CONTROL_∞(METAnG, $\mathscr{L}$) so that the iteration of the linear control operator (or of HOM($\mathscr{L}$)) coincides with the iteration of the metalinear control operator. It is not just the metalinear control operator which has this property. The same results holds for the iteration of controls on all nonterminal bounded grammars. We shall give the more general result in the next section.

7. Nonterminal Bounded Grammars, LDB Grammars, and Homomorphic Replication

In this section we show that for any nonterminal bounded grammar G and any full semiAFL $\mathscr{L}$, the closure of $\mathscr{L}$ under the G-control operator is contained in the closure of $\mathscr{L}$ under the linear-control operator, i.e., under homomorphic replication. We show that this property really characterizes nonterminal bounded grammars in the sense that $\text{CONTROL}_\infty(\mathscr{G}(G), \mathscr{L})$ is contained in $\text{CONTROL}_\infty(\text{LINEARG}, \mathscr{L})$ for every full semiAFL $\mathscr{L}$ if and only if G is c-equivalent to a nonterminal bounded grammar and that $\text{CONTROL}_\infty(\mathscr{G}(G), \mathscr{L}) = \text{CONTROL}_\infty(\text{LINEARG}, \mathscr{L})$ for every full semiAFL if and only if G is self-embedding and c-equivalent to a nonterminal bounded grammar. Then we use this property to obtain the same hierarchy theorems for general nonterminal bounded control operators as for the linear control operator. Finally we give a characterization of $\text{CONTROL}_\infty(\text{LDBG}, \mathscr{L})$ in terms of an infinite homomorphic replication operator, and give a weaker hierarchy theorem for this case.

Let us start by reviewing the formal definition of a nonterminal bounded grammar [2].

Definition. A grammar G is *nonterminal bounded of degree* r for $r \geqslant 1$ if for each nonterminal X in G, whenever $X \overset{*}{\Rightarrow} u$ then u contains at most r nonterminals. Let NTBrG be the family of nonterminal bounded grammars of degree r and let NTBrL be the corresponding family of languages.

Theorem 7.1. *For any* semiAFL $\mathscr{L}$,

$$\text{CONTROL}(\text{NTBnG}, \mathscr{L}) \subseteq \text{CONTROL}_n(\text{LINEARG}, \mathscr{L})$$

whenever $1 \leqslant n \leqslant 2^{m-1}$.

Proof. Let $\mathscr{L}$ be a semiAFL. By definition, for $r \leqslant n$, NTBrG $\subseteq$ NTBnG and so $\text{CONTROL}(\text{NTBrG}, \mathscr{L}) \subseteq \text{CONTROL}(\text{NTBnG}, \mathscr{L})$. Hence we may as well assume that $n = 2^{m-1}$.

Notice that for any language A, the m-fold iteration of ρ on A is

$$\rho^m(A) = \{(ww^R)^{2^{m-1}} \mid w \in A\}$$

and is in $\text{CONTROL}_m(\text{LINEARG}, \mathscr{L})$ by Theorem 6.1. When discussing a word $(ww^R)^{2^{m-1}}$ in a language of the form $\rho^m(A)$, we shall call an occurrence of ww^R a *block*, with w the *left semiblock* and w^R *the right semiblock*.

Let $G = (V, \Sigma, P, S)$ be a nonterminal bounded grammar of degree n. We can assume that rules of G are of the forms $Z \to XY$, $Z \to \alpha Y \beta$, and $Z \to \alpha$ for $\alpha, \beta \in \Sigma^*$, $X, Y \in V - \Sigma$. (For otherwise we can replace a rule $Z \to u_1 Y_1 \cdots u_r Y_r u_{r+1}$, $r \geqslant 2$, the u_i terminal strings and the Y_i nonterminals, by a rule set $\{Z \to u_1[Y_1 \cdots Y_r]\, u_{r+1}$, $[Y_1 \cdots u_r Y_r] \to [Y_1 \cdots u_r]Y_r$, $[Y_1 \cdots Y_{r-1}u_r] \to [Y_1 \cdots Y_{r-1}]u_r$, ..., $[Y_1 u_2 Y_2] \to [Y_1 u_2]Y_2$, $[Y_1 u_2] \to Y_1 u_2\}$ for appropriate new nonterminals of the form $[v]$ and then apply Proposition 2.8 to the new grammar to obtain c-equivalence.) Let C be a control set in $\mathscr{L}$. We define an auxiliary grammar G', homomorphisms h_1 and h_2, and special points in a word of C as follows. Let $\$$ and $\not{\epsilon}$ be new symbols. For a rule $p\colon Z \to u$ in G, u in Σ^*, we

call p an *a-point*, put into G' a corresponding rule $p: Z \to e$ and set $h_1(p) = u$, $h_2(p) = e$. For a rule $p: Z \to XY$ in G, X and Y nonterminals, we call p a *b-point*, put into G' a corresponding rule $p: Z \to XY\not{\phi}$ and set $h_1(p) = h_2(p) = e$. For a rule $p: Z \to uYv$ in G, u, v in Σ^* and Y a nonterminal, we let G' have a corresponding rule $p: Z \to Y$ and set $h_1(p) = u$, $h_2(p) = v$. We call both a-points and b-points the *critical points*. Notice that since G is in NTBnG, G' will never generate a word longer than $2n - 1$, corresponding to at most n a-points and at most $n - 1$ b-points in any derivation of G. If we divide a control word in C into upv for a critical point p, and $S \Rightarrow^{up} z$ in G', we call z the *stack* corresponding to critical point p. We let the p's in $upvv^Rpu^R$ be *corresponding* critical points.

Let $A = \rho^m(\$C\$)$. We shall construct an a-transducer M such that $L(G, C) = M(A)$ and then the desired conclusion follows from Theorem 6.1. The idea is that in one scan through a control word w, M can record in its finite state control the sequence of critical points and the stack contents at each critical point. Scanning through the successive blocks of $(\$w\$\$w^R\$)^{2^{m-1}}$, machine M can use h_1 and h_2 to output the strings generated by G. For example, in a subderivation $X \overset{*}{\Rightarrow} uXv$, M can use the left semiblock w to output u and the right semiblock w^R to output v. However if the derivation proceeds $uXv \Rightarrow uYZv$, then M will have to wait for a later block to output v, after the derivation from Z has been simulated. But, notice that the b-point corresponding to $X \to YZ$ appears in w before all portions of the control word applying to Z and hence the corresponding b-point in the right semiblock $\$w^R\$$ must appear after the portion concerned with the derivation from Z. Thus we will be able to pick up v from the right semiblock in the same block in which we complete the derivation from Z. So we need only as many blocks as a-points and the maximum number of a-points is $n = 2^{m-1}$. Hence we have enough blocks in the words of A to go around.

Let us outline briefly the action of M on $(\$w\$\$w^R\$)^n$ for a control word w in C. In its first scan through w, M will record in its finite state control the succession of up to $2n - 1$ critical points and the corresponding stacks. If w does not control a completed derivation in G, M halts without output.

Machine M has a special register called the *stack register* which initially has contents S. It also stores an integer 0 to n called the *a-number*. Initially the a-number is 0; by convention we let the 0th a-point in $\$w$ be $\$$ itself. All this is stored in the finite state control.

If M ever ends a block with empty stack register and a-number equal to the number of a-points in w (indicating that it has simulated completely the derivation) then M enters an accepting state, scans through the rest of its input without further printing and halts giving output.

Now consider the action of M at the start of a block $\$w\$\$w^R\$$ with a-number i. The stack register will contain the stack corresponding to the ith a-point of w and this stack will be of the form Xy for some nonterminal X.

The action of M can be divided into three steps. Steps 1 and 2 always occur, while Step 3 may not occur or may be repeated.

Step 1. First M scans through the left semiblock without printing until it reaches the ith a-point. Now M scans through the section of w from just right of the ith a-point up

to and including the $(i+1)$th a-point printing the image of this section under h_1; if this segment is u, then M prints $h_1(u)$. Next M scans without writing the rest of the left semiblock and proceeds to Step 2 on entering the right semiblock $\$w^R\$$.

Step 2. Now M continues to scan without printing until it reaches the a-point corresponding to the $(i+1)$th a-point of $\$w\$$. Then it scans until the next critical point whether that be an a-point or b-point, using h_2 to print; thus if this segment is v, M prints $h_2(v)$ and v is of course an initial segment of u^R. Next M changes the a-number to $i+1$ and the stack register contents to the stack corresponding to the $(i+1)$th a-point of w. If the stack is now of the form $\not{c}z$, M proceeds to Step 3. Otherwise, M scans to the end of the right semiblock without printing.

Step 3. At the start of this step, M always has stack register content $\not{c}z$ for appropriate z and is within the right semiblock. This occurrence of $\not{c}$ arose from a rule $X \to YZ\not{c}$ in G' and $X \to YZ$ in G which was used at some b-point in w, say the jth b-point. The crucial point is that, as mentioned above, this b-point must occur either where M now sits or to the right because it occurred earlier in the derivation and hence there or to the left in the left semiblock, $\$w\$$. Hence M can scan the right semiblock without printing until it reaches the critical point corresponding to the jth b-point in w. It then uses h_2 to print until the next critical point is reached. Now M changes the stack register to z and will either repeat Step 3 if z is itself of the form $\not{c}z'$ or else will scan without printing the rest of the block, ending with stack register z and a-number still $i+1$.

If M ever reaches the end of its input with nonempty stack register or a-number less than the total number of a-points in w, it halts in a nonaccepting state and gives no output.

So $L(G, C) = M(A) = M(\rho^m(\$C\$)) \in \text{CONTROL}_m(\text{LINEARG}, \mathscr{L})$. ∎

The relationship between n and m cannot be improved in general. For suppose $\mathscr{L}$ is a full semiAFL closed under concatenation but not under the linear-control operator. There is an e-free language L in $\mathscr{L}$ such that $\rho(L)$ is not in $\mathscr{L}$. Let c be a new symbol. For each $m \geqslant 0$, let $L_m = (\rho(L)c)^{2^m}$. By Lemma 6.9, L_m is not in $\text{CONTROL}_m(\text{LINEARG}, \mathscr{L})$. However, L_m is in $\text{CONTROL}(\text{METAnG}, \mathscr{L})$ for $n = 2^m$ and thus in $\text{CONTROL}_{m+1}(\text{LINEARG}, \mathscr{L})$.

We now observe that Theorem 7.1 cannot be improved by significantly reducing the restrictions on the class of grammars involved. In fact, $\text{CONTROL}(\mathscr{G}(G), \mathscr{L}) \subseteq \text{CONTROL}_\infty(\text{LINEARG}, \mathscr{L})$ for every full semiAFL $\mathscr{L}$ if and only if G is c-equivalent to a nonterminal bounded grammar.

DEFINITION. Let G be a left derivation bounded grammar. We call G *quasi-nonterminal bounded* if whenever $X \overset{*}{\Rightarrow} uXv$ in G, u does not contain any right self-embedding symbols, and v is a terminal string.

LEMMA 7.2. *If G is quasi-nonterminal bounded it is c-equivalent to a nonterminal bounded grammar.*

Proof. Let $G = (V, \Sigma, P, S)$ be quasi-nonterminal bounded. If G is trivial it is obviously c-equivalent to a trivial regular grammar. Assume G is nontrivial. Since G is

ldb we can, using Proposition 2.8, transform it into a strongly pse grammar which is also quasi-nonterminal bounded. So assume that G is strongly pse.

Let I be the subset of $V - \Sigma$ containing all right self-embedding symbols. Let $t = \text{Max}(\{2\} \cup \{| y | \mid Z \to y \text{ is in } P\})$ and $n = (\#(V - \Sigma) + 1)t$. We shall use the fact that if $X \Rightarrow^{L^*} \alpha Yv Zu$ for a terminal string α and nonterminals Y and Z using a derivation in which no right self-embedding symbols were used to produce YvZ (except possibly for Z itself), then $| YvZ | \leqslant n$.

We shall define a grammar G' which is nonterminal bounded and is a transform of G. For each v in $V^+ - \Sigma^+$ with $1 \leqslant | v | \leqslant n$, let $[v]$ be a new symbol and let V' contain all such symbols plus all of Σ. We associate to each rule p in P a set $S(p)$ of rules as follows.

For a rule

$$p\colon X \to u_1 Y_1 \cdots u_r Y_r u_{r+1} \qquad Y_1, \ldots, Y_r \in I,\ r \geqslant 1,\ u_i \in (V - I)^*$$

$S(p)$ contains all possible rules of the form

$$[\alpha X v \beta] \to \alpha[u_1 Y_1] \cdots [u_r Y_r][u_{r+1} v]\beta$$

for $| \alpha X v \beta | \leqslant n$, $| u_{r+1} v | \leqslant n$, $\alpha, \beta \in \Sigma^*$, $v \in V^* - V^*\Sigma$ and either $v \neq e$ or u_{r+1} not in Σ^*, and all possible rules of the form

$$[\alpha X \beta] \to \alpha[u_1 Y_1] \cdots [u_r Y_r]\, u_{r+1}\beta$$

for $| \alpha X \beta | \leqslant n$, $\alpha, \beta \in \Sigma^*$, and $u_{r+1} \in \Sigma^*$.

For a rule

$$p\colon X \to \delta u \quad \delta \in \Sigma^*, \qquad u \in (V - I)^* - \Sigma V^*$$

$S(p)$ contains all possible rules of the form

$$[\alpha X v \beta] \to \alpha\delta[uv]\beta$$

for $| \alpha X v \beta | \leqslant n$, $| uv | \leqslant n$, $\alpha, \beta \in \Sigma^*$, $v \in V^* - V^*\Sigma$, and $uv \neq e$ and all possible rules of the form

$$[\alpha X \beta] \to \alpha\delta\beta$$

for $| \alpha X \beta | \leqslant n$, $u = e$, and $\alpha, \beta \in \Sigma^*$.

Let $P' = \bigcup_{p \in P} S(p)$ and $G' = (V', \Sigma, P, [S])$. First we wish to argue that G' is nonterminal bounded. This is equivalent to showing that if $Z \overset{*}{\Rightarrow} uZv$ then u does not contain nonterminals. We claim that we cannot have in G' a derivation of the form

$$[v] \overset{*}{\Rightarrow} —[u]—[—v—]—.$$

(In order to save subscripting, we use "—" to indicate some possibly empty string whose identity is unimportant to the argument.) Suppose v is the shortest string for which this situation occurs. There are two cases. If $v = Z$ for a nonterminal Z, we observe by studying rules in P' that in generating $[u]$, Z must first have produced some symbol of the

form $[u_1Xu_2]$ for X a right self-embedding symbol (otherwise any string from $[Z]$ would contain at most one nonterminal). Thus we can rearrange the derivation to obtain

$$[Z] \overset{*}{\Rightarrow} —[u_1Xu_2]—[—Z—]—.$$

But this must correspond to a derivation in G

$$Z \overset{*}{\Rightarrow} —u_1Xu_2— -Z- —$$

for X right self-embedding, which violates the definition of quasi-nonterminal bounded. Hence we cannot have $|v| = 1$. Suppose $v = Zv'$ for a nonterminal Z and $v' \neq e$. Thus we have

$$[Zv'] \overset{*}{\Rightarrow} —[u]—[—Zv'—]—$$

and in G

$$Zv' \overset{*}{\Rightarrow} —u——Zv'——.$$

An initial part of the string is generated from Z and the rest from v'. If Z generates the initial substring through Z then we can obtain the same contradiction as before. Thus v' generates at least the part from Zv' on and perhaps more. So in G' we have a derivation of the form

$$[v'] \overset{*}{\Rightarrow} —[—Zv'—]—.$$

But we can continue this derivation to get one of the form

$$[v'] \overset{*}{\Rightarrow} —[—Zv'—]— \overset{*}{\Rightarrow} —[u]—[—Zv'—]—$$

which of course violates the minimality of v. Hence G' is nonterminal bounded.

Now we must argue that G is c-equivalent to G'. First notice that if we consider productions to be their own names we can view $S(p)$ as a finite substitution and for any control set C on G we have $L(G, C) = L(G', S(C))$. Thus for any semiAFL $\mathscr{L}$,

$$\text{CONTROL}(\{G\}, \mathscr{L}) \subseteq \text{CONTROL}(\{G'\}, \mathscr{L}).$$

Let $H = \sigma(G)$ be a complete interpretation of G. By Proposition 2.13, since G is left derivation bounded we can assume that σ is *simple*, i.e., $\sigma(Z) = \{Z\}$ for every nonterminal Z. Let σ' be the complete interpretation of G' defined by $\sigma'(a) = \sigma(a)$ for every terminal a and $\sigma'([v]) = \{[v'] \mid v' \in \sigma(v)\}$ for every nonterminal $[v]$ in G'. Let $H' = \sigma'(G')$. Then the relationship between H and H' is similar to the relationship between G and G'. In particular we can defined a substitution S' from rules of H to those of H'; if a rule p in H is in $\sigma(p_1)$ for p_1 in G, then $S'(p)$ consists of those rules of $\sigma'(S(p_1))$ which have the "right terminals" in the "right places." Again $L(H, C) = L(H', S'(C))$ for any control set C. So we have $G \leqslant^c G'$.

For any rule p in P' there is a unique p_1 in P such that p is in $S(p_1)$; let $g(p) = p_1$. We can regard g as a homomorphism from names of rules in P' to names of rules in P. For any control set C, $L(G', C) = L(G, g(C))$ so for any semiAFL $\mathscr{L}$, CONTROL $(\{G'\}, \mathscr{L}) \subseteq$ CONTROL$(\{G\}, \mathscr{L})$.

Now let $H' = \sigma'(G')$ be any complete interpretation of G'. As before we can assume that σ' is simple. We define σ as the corresponding simple interpretation of G, i.e., $\sigma(a) = \sigma'(a)$ for every terminal a and $\sigma(Z) = \{Z\}$ for every terminal Z, and let $H = \sigma(G)$. Unfortunately we cannot use a simple correspondence between rules of H' and H. The problem is that while in G' we would only have rules like $[\alpha Zv\beta] \to \alpha y\beta$, now in H' we can have $[\alpha Zv\beta] \to \alpha' y\beta'$ for terminal strings α' in $\sigma'(\alpha)$ and β' in $\sigma'(\beta)$. But in H the identities of α' and β' have already been determined before the corresponding rule can be applied. However H and H' are ldb so we can make appropriate guesses and verify and record our guesses. We describe an a-transducer M such that $L(H', C) = L(H, M(C))$ for any control set C. The point is that M can record in its finite state control the sequence of nonterminals in the current derivation in H'. Consider a rule of H' of the form $[\alpha Zv\beta] \to \alpha'[y_1] \cdots [y_r][y_{r+1}v]\,\beta'$. Now M will already have recorded its guess as to the proper identity of α' and β'. If these guesses were wrong it blocks. The corresponding rule in G looks like $Z \to y_1 \cdots y_r\, y_{r+1}$ and now M must output a rule in $\sigma(Z \to y_1 \cdots y_{r+1})$. It selects one at random. Now it can erase $[\alpha Zv\beta]$ from its memory and replace it with $[y_1] \cdots [y_r][y_{r+1}v]$. However, if any y_i contains terminals then in selecting output M has guessed which member of $\sigma(y_i)$ is "right" and so it records this guess along with $[y_i]$ itself. It has already guessed at $\sigma(v)$ and so carries this guess along as part of the guess for $\sigma(y_{r+1}v)$. The behavior of M on the other types of rules of H' can be similarly described. Thus we can build M so that $L(H', C) = L(H, M(C))$ for any control set C. Hence $G' \leqslant^c G$ and we are done. ∎

We have seen that $\text{CONTROL}_\infty(\mathscr{G}(G), \mathscr{L}) \subseteq \text{CONTROL}_\infty(\text{LINEARG}, \mathscr{L})$ when G is nonterminal bounded. We now observe that this condition on G is necessary as well as sufficient.

THEOREM 7.3. $\text{CONTROL}_\infty(\mathscr{G}(G), \mathscr{L}) \subseteq \text{CONTROL}_\infty(\text{LINEARG}, \mathscr{L})$ *for every full* semiAFL $\mathscr{L}$ *if and only if G is c-equivalent to a nonterminal bounded grammar. Further,* $\text{CONTROL}_\infty(\mathscr{G}(G), \mathscr{L}) = \text{CONTROL}_\infty(\text{LINEARG}, \mathscr{L})$ *for every full* semiAFL *if and only if G is c-equivalent to a self-embedding nonterminal bounded grammar.*

Proof. If G is c-equivalent to a nonterminal bounded grammar G' then $\text{CONTROL}_\infty(\mathscr{G}(G), \mathscr{L}) = \text{CONTROL}_\infty(\mathscr{G}(G'), \mathscr{L}) \subseteq \text{CONTROL}_\infty(\text{LINEARG}, \mathscr{L})$ for every full semiAFL $\mathscr{L}$ by Theorem 7.1. If G' is also self-embedding then by the usual arguments we can form some c-equivalent grammar G'' by adding appropriate rules $S \to X$, $X \to aXa$, and $X \to a$ and every linear context-free grammar will be c-equivalent to some member of $\mathscr{G}(G'')$. Thus $\text{CONTROL}_\infty(\text{LINEARG}, \mathscr{L}) \subseteq \text{CONTROL}_\infty(\mathscr{G}(G), \mathscr{L}) \subseteq \text{CONTROL}_\infty(\text{LINEARG}, \mathscr{L})$ for every full semiAFL $\mathscr{L}$.

Now suppose on the contrary that $\text{CONTROL}(\mathscr{G}(G), \mathscr{L}) \subseteq \text{CONTROL}_\infty(\text{LINEARG}, \mathscr{L})$ for every full semiAFL $\mathscr{L}$. We can assume that G is reduced. If G is not c-equivalent to an ldb grammar, then $\text{CONTROL}(\mathscr{G}(G), \text{LINEARL}) = \text{RE}$ whereas $\text{CONTROL}_\infty(\text{LINEARG}, \text{LINEARL})$ is a proper subset of the context-sensitive languages. So we may as well assume that G is ldb. We claim that G must be quasi-nonterminal bounded and hence the desired result follows from Lemma 7.2. For suppose $X \overset{+}{\Rightarrow} uZvXy$ and Z is right self-embedding. Clearly X is left self-embedding. By the usual arguments G is

c-equivalent to the grammar G_1 formed by adding production set $\{S \to X, X \to ZX, Z \to Za, X \to aX, X \to e, Z \to e\}$ to G.

We shall show that if $\mathscr{L}$ is any full AFL not closed under reversal (e.g. the full AFL generated by $\{a^n b^m \mid 1 \leqslant n \leqslant m\}$) then CONTROL($\mathscr{G}(G_1)$, $\mathscr{L}$) contains languages not in CONTROL$_\infty$(LINEARG, $\mathscr{L}$) which yields the desired contradiction. So let $\mathscr{L}$ be a full AFL not closed under reversal. There is a language $L \subseteq \Sigma^+$ for some alphabet Σ such that L^R is not in $\mathscr{L}$. Let c and d be new symbols. Then LcL^R is not in $\mathscr{L}$ or in $\mathscr{L}^R$ and hence $(LcL^Rd)^+$ is not in CONTROL$_\infty$(LINEARG, $\mathscr{L}$). We shall show that it is in CONTROL($\mathscr{G}(G_1)$, $\mathscr{L}$). Let h be a length preserving homomorphism which gives each member of $\Sigma \cup \{c, d\}$ a new name. Let ϕ_1, ϕ_2, $\mathscr{S}_1$, and $\mathscr{S}_2$ be new symbols. Let G_2 be the grammar in $\mathscr{G}(G_1)$ with rules $S \to X$ labeled ϕ_1, $X \to ZX$ labeled ϕ_2, $Z \to e$ labeled $\mathscr{S}_1$, and $X \to e$ labeled $\mathscr{S}_2$ and for each a in $\Sigma \cup \{c, d\}$ rules $X \to aX$ labeled a and $Z \to Za$ labeled $h(a)$. Now the language $L_1 = \phi_1(Lc\phi_2 h(dL)\mathscr{S}_1)^+\mathscr{S}_2$ is in $\mathscr{L}$ so $L(G_2, L_1) = (LcL^Rd)^+$ is in CONTROL($\mathscr{G}(G_1)$, $\mathscr{L}$) = CONTROL($\mathscr{G}(G)$, $\mathscr{L}$).

Thus G is quasi-nonterminal bounded and so c-equivalent to a nonterminal bounded grammar. Now suppose that additionally CONTROL$_\infty$($\mathscr{G}(G)$, $\mathscr{L}$) = CONTROL$_\infty$ (LINEARG, $\mathscr{L}$) for every full semiAFL $\mathscr{L}$. If G is not self-embedding then CONTROL ($\mathscr{G}(G)$, REGULARL) = $\mathscr{L}(G)$ = REGULARL so CONTROL$_\infty$($\mathscr{G}(G)$, REGULARL)= REGULARL while CONTROL(LINEARG, REGULARL) = LINEARL. Hence G must be self-embedding. ∎

We can use Theorem 7.1 to establish analogs of Theorems 6.10, 6.11, and 6.13. The general idea is that if G is self-embedding and nonterminal bounded of degree n then CONTROL$_k$($\mathscr{G}(G)$, $\mathscr{L}$) forms an increasing chain of families of languages which interweaves with the chain CONTROL$_k$(LINEARG, $\mathscr{L}$) by the formula CONTROL$_k$ (LINEARG, $\mathscr{L}$) $\subseteq$ CONTROL$_k$($\mathscr{G}(G)$, $\mathscr{L}$) $\subseteq$ CONTROL$_{nk}$(LINEARG, $\mathscr{L}$). Thus if $\mathscr{L}$ is a full semiAFL not closed under the G-control operator it is also not closed under the linear-control operator and we know that the latter gives us a hierarchy. Hence the G-control operator gives us a possibly different hierarchy converging to the same bound,

$$\text{Hom}_\infty(\mathscr{L}) = \text{CONTROL}_\infty(\text{LINEARG}, \mathscr{L}) = \text{CONTROL}_\infty(\mathscr{G}(G), \mathscr{L}).$$

First, Corollary 6.7 becomes the following.

THEOREM 7.4. *For any self-embedding nonterminal bounded grammar G and any full* semiAFL $\mathscr{L}$, *if* $\mathscr{L} \neq$ CONTROL($\mathscr{G}(G)$, $\mathscr{L}$) *then for all* $k \geqslant 0$

$$\text{CONTROL}_k(\mathscr{G}(G), \mathscr{L}) \neq \text{CONTROL}_{k+1}(\mathscr{G}(G), \mathscr{L}).$$

Proof. First observe that $\mathscr{L} \neq$ CONTROL(LINEARG, $\mathscr{L}$) since otherwise $\mathscr{L}$ = CONTROL(LINEARG, $\mathscr{L}$) = CONTROL$_\infty$(LINEARG, $\mathscr{L}$) = CONTROL$_\infty$($\mathscr{G}(G)$, $\mathscr{L}$) and hence $\mathscr{L}$ = CONTROL($\mathscr{G}(G)$, $\mathscr{L}$) contrary to hypothesis. If G is nonterminal bounded of degree n and CONTROL$_k$($\mathscr{G}(G)$, $\mathscr{L}$) = CONTROL$_{k+1}$($\mathscr{G}(G)$, $\mathscr{L}$), then again CONTROL$_\infty$(LINEARG, $\mathscr{L}$) = CONTROL$_\infty$($\mathscr{G}(G)$, $\mathscr{L}$) = CONTROL$_k$ ($\mathscr{G}(G)$, $\mathscr{L}$) $\subseteq$ CONTROL$_{nk}$(LINEARG, $\mathscr{L}$) by Theorem 7.1, so in particular CON-

$\text{TROL}_{nk}(\text{LINEARG}, \mathscr{L}) = \text{CONTROL}_{nk+1}(\text{LINEARG}, \mathscr{L})$ a contradiction of Corollary 6.7. ▮

Using Theorem 7.1 the following consequence of Theorem 6.10 is immediate.

THEOREM 7.5. *Let $\mathscr{L}$ be a full* semiAFL *and G a self-embedding nonterminal bounded grammar.*

(a) *If $\mathscr{L}$ is not closed under Kleene+ (i.e., is not a full* AFL*), then* $\text{CONTROL}_\infty(\mathscr{G}(G), \mathscr{L})$ *does not contain the closure of $\mathscr{L}$ under Kleene+ and hence is not closed under Kleene+.*

(b) *If $\mathscr{L}$ is not closed under concatenation, then for all $k \geqslant 0$,* $\text{CONTROL}_k(\mathscr{G}(G), \mathscr{L})$ *does not contain the closure of $\mathscr{L}$ under concatenation and hence is not closed under concatenation and* $\text{CONTROL}_\infty(\mathscr{G}(G), \mathscr{L})$ *does not contain the closure of $\mathscr{L}$ under Kleene+ and hence is not closed under Kleene+.*

Now we have the analog of Theorem 6.11.

THEOREM 7.6. *Let G be a self-embedding nonterminal bounded grammar. If $\mathscr{L}$ is a full* semiAFL *such that $\mathscr{L} \neq \text{CONTROL}(\mathscr{G}(G), \mathscr{L})$ then for each $k \geqslant 1$,* $\text{CONTROL}_k(\mathscr{G}(G), \mathscr{L})$ *is not closed under concatenation and* $\text{CONTROL}_\infty(\mathscr{G}(G), \mathscr{L})$ *is not closed under Kleene+.*

Proof. In view of Theorem 7.5, we need only show that if $\mathscr{L} \neq \text{CONTROL}(\mathscr{G}(G), \mathscr{L})$ for a full semiAFL $\mathscr{L}$, then $\text{CONTROL}(\mathscr{G}(G), \mathscr{L})$ is not closed under concatenation. As in the proof of Theorem 7.4 we can conclude that $\mathscr{L} \neq \text{CONTROL}(\text{LINEARG}, \mathscr{L})$ and so by Theorems 6.10 and 6.11 the closure under concatenation of CONTROL (LINEARG, $\mathscr{L}$) cannot be contained in any family $\text{CONTROL}_k(\text{LINEARG}, \mathscr{L})$. But if $\text{CONTROL}(\mathscr{G}(G), \mathscr{L})$ is closed under concatenation it contains the concatenation closure of CONTROL(LINEARG, $\mathscr{L}$) and if G is nonterminal bounded of degree n, so does $\text{CONTROL}_n(\text{LINEARG}, \mathscr{L})$, a contradiction. ▮

An analog of Lemma 6.12 would not hold in all cases. For example, let $\mathscr{L}_1 = \text{REGULARL}$ and $\mathscr{L}_2 = \text{LINEARL}$. Then $\text{CONTROL}(\text{META2G}, \mathscr{L}_1) = \text{META2L}$ which certainly properly contains $\mathscr{L}_2 = \text{LINEARL}$ [18]. But META2L is the closure under union of all languages expressible as the concatenation of two linear context-free languages and so $\text{CONTROL}(\text{META2G}, \mathscr{L}_1)$ is a fortiori contained in $\mathscr{F}(\mathscr{L}_2)$. However, we do have the following result for the special case where $\mathscr{L}_1 = \mathscr{L}_2$.

LEMMA 7.7. *Let G be a self-embedding nonterminal bounded grammar. For any full* semiAFL $\mathscr{L}$,

$$\text{CONTROL}(\mathscr{G}(G), \mathscr{L}) \subseteq \mathscr{F}(\mathscr{L}) \quad \text{if and only if } \text{CONTROL}(\mathscr{G}(G), \mathscr{L}) \subseteq \mathscr{L}.$$

Proof. Suppose $\text{CONTROL}(\mathscr{G}(G), \mathscr{L}) \subseteq \mathscr{F}(\mathscr{L})$. Then CONTROL(LINEARG, $\mathscr{L}$) is also contained in $\mathscr{F}(\mathscr{L})$ so by Lemma 6.12, CONTROL(LINEARG, $\mathscr{L}$) is contained

in $\mathscr{L}$. Thus $\mathscr{L}$ is closed under the linear-control operator and $\text{CONTROL}_\infty(\mathscr{G}(G), \mathscr{L}) = \text{CONTROL}_\infty(\text{LINEARG}, \mathscr{L}) \subseteq \mathscr{L}$. ∎

Finally, we can obtain the appropriate version of Theorem 6.13.

THEOREM 7.8. *Let G be a self-embedding nonterminal bounded grammar. Let $\mathscr{L}$ be a full* semiAFL *such that* $\mathscr{L} \neq \text{CONTROL}(\mathscr{G}(G), \mathscr{L})$.

(1) *For all* $k \geqslant 0$

(a) $\text{CONTROL}_{k+1}(\mathscr{G}(G), \mathscr{L}) - \mathscr{F}(\text{CONTROL}_k(\mathscr{G}(G), \mathscr{L})) \neq \phi$,

and

(b) $\text{CONTROL}_k(\mathscr{G}(G), \mathscr{L})$ *does not contain the closure under concatenation of* $\text{CONTROL}(\text{LINEARG}, \mathscr{L})$.

(2) *For all* $k \geqslant 1$

$$\mathscr{F}(\text{CONTROL}_k(\mathscr{G}(G), \mathscr{L})) - \text{CONTROL}_{k+1}(\mathscr{G}(G), \mathscr{L}) \neq \phi.$$

(3) $\mathscr{F}(\text{CONTROL}(\mathscr{G}(G), \mathscr{L}))$ *and* $\text{CONTROL}_\infty(\mathscr{G}(G), \mathscr{L})$ *are incomparable.*

(4) *If* $\mathscr{L} \neq \mathscr{F}(\mathscr{L})$ *then* $\mathscr{F}(\mathscr{L})$ *and* $\text{CONTROL}_\infty(\mathscr{G}(G), \mathscr{L})$ *are incomparable.*

Proof. By Theorem 7.4 $\text{CONTROL}_{k+1}(\mathscr{G}(G), \mathscr{L}) - \text{CONTROL}_k(\mathscr{G}(G), \mathscr{L}) \neq \phi$, so (1a) follows from this inequality by Lemma 7.1 applied to $\text{CONTROL}_k(\mathscr{G}(G), \mathscr{L})$. Suppose G is nonterminal bounded of degree n. As before $\mathscr{L} \neq \text{CONTROL}(\text{LINEARG}, \mathscr{L})$ so by Theorem 6.1 $\text{CONTROL}_{nk}(\text{LINEARG}, \mathscr{L})$ does not contain the closure under concatenation of $\text{CONTROL}(\text{LINEARG}, \mathscr{L})$ and by Theorem 7.1 neither does $\text{CONTROL}_k(\mathscr{G}(G), \mathscr{L})$. This establishes (1b). Statements (2) and (3) are now immediate. Statement (4) is identical to the same statement in Theorem 6.13 since $\text{CONTROL}_\infty(\mathscr{G}(G), \mathscr{L}) = \text{CONTROL}_\infty(\text{LINEARG}, \mathscr{L})$. ∎

We conclude this section by giving a general characterization result for controls on left derivation bounded grammars. Closure under the control operator corresponding to nonterminal bounded grammars is equivalent to closure under homomorphic replication. We now show that for left derivation bounded grammars we need a more general type of homomorphic replication. Instead of considering $f_{n,c}$ for fixed n, we examine $f_{\infty,c}$, the union of all the $f_{n,c}$.

DEFINITION. If $L \subseteq (\Sigma \cup \{c\})^*$ for a finite vocabulary Σ and a special symbol c not in Σ and h_1 and h_2 are homomorphisms, let

$$f_{\infty,c}(h_1, h_2, L) = \bigcup_{n \geqslant 1} f_{n,c}(h_1, h_2, L)$$
$$= \{h_1(w_1)\, h_2(w_1{}^R) \cdots h_1(w_n)\, h_2(w_n{}^R) \mid n \geqslant 1, w_1 c \cdots w_n c \in L, w_1, \ldots, w_n \in \Sigma^*\}.$$

DEFINITION. For a family of languages $\mathscr{L}$, let

$$\text{IHOM}(\mathscr{L}) = \{f_{\infty,c}(h_1, h_2, L) \mid L \in \mathscr{L}, h_1, h_2 \text{ homomorphisms}\}.$$

Let $\mathrm{IHOM}_0(\mathscr{L}) = \mathscr{L}$ and for $k \geqslant 0$, let

$$\mathrm{IHOM}_{k+1}(\mathscr{L}) = \mathrm{IHOM}(\mathrm{IHOM}_k(\mathscr{L})).$$

Let $\mathrm{IHOM}_\infty(\mathscr{L}) = \bigcup_{n \geqslant 0} \mathrm{IHOM}_n(\mathscr{L})$.

DEFINITION. Let LDBnG be the family of left derivation bounded grammars with left derivation bound n.

We note without proof the following.

LEMMA 7.9. *Let $\mathscr{L}$ be a* semiAFL *and let G be the left derivation bounded grammar* $G = (\{S, A, a\}, \{a\}, \{S \to AS, S \to e, A \to aAa, A \to e\}, S)$. *Then* $\mathrm{IHOM}(\mathscr{L}) = \mathrm{CONTROL}(\mathscr{G}(G), \mathscr{L})$ *and so* $\mathrm{IHOM}(\mathscr{L})$ *is a full* semiAFL *contained in* $\mathrm{CONTROL}(\mathrm{LDB2G}, \mathscr{L})$ *and containing* $\bigcup_k \mathrm{CONTROL}(\mathrm{METAkG}, \mathscr{L})$. *If $\mathscr{L}$ is closed under concatenation, so is* $\mathrm{IHOM}(\mathscr{L})$ *and if $\mathscr{L}$ is a full* AFL *so is* $\mathrm{IHOM}(\mathscr{L})$. *If $\mathscr{L}$ is a full* AFL *and* $\mathscr{L} \neq \mathrm{CONTROL}(\mathrm{LINEARG}, \mathscr{L})$, *then* $\mathrm{IHOM}(\mathscr{L}) - \mathrm{CONTROL}_\infty(\mathrm{LINEARG}, \mathscr{L}) \neq \phi$.

Our main result in this section is that $\mathrm{IHOM}_\infty(\mathscr{L})$ and $\mathrm{CONTROL}_\infty(\mathrm{LDBnG}, \mathscr{L})$ coincide for any semiAFL $\mathscr{L}$ and any $n \geqslant 2$.

THEOREM 7.10. *For any* semiAFL $\mathscr{L}$, *and any* $n \geqslant 2$,

$$\mathrm{CONTROL}(\mathrm{LDBnG}, \mathscr{L}) \subseteq \mathrm{IHOM}_n(\mathscr{L}).$$

Proof. We shall actually prove something stronger. Let $G = (V, \Sigma, P, S)$ be left derivation bounded. Call a symbol X in G *finite state* if whenever $X \overset{*}{\Rightarrow} u$, then u is in $\Sigma^*(V \cup \{e\})$. Let I be the set of finite state symbols in G. Let

$$m(G) = \mathrm{Max}(\{m \mid \exists u \in (\Sigma \cup I)^*, v \in V^*, Y \in V - \Sigma - I, S \overset{L^*}{\Rightarrow} uYv \text{ and } Yv \text{ contains } m \text{ nonterminals}\}).$$

In computing members of $L(G, C)$ one need worry only about nonterminals to the right of the leftmost nonterminal which is not finite state. Initial strings of finite state symbols can be ignored in determining how many replications of the IHOM operator are needed.

We shall show by induction on $m(G)$ that

(*) $\mathrm{CONTROL}(\{G\}, \mathscr{L}) \subseteq \mathrm{IHOM}_{m(G)}(\mathscr{L})$.

For $m(G) = 0$, all nonterminals are finite state so G is regular and $\mathrm{CONTROL}(\{G\}, \mathscr{L}) \subseteq \mathscr{L} = \mathrm{IHOM}_0(\mathscr{L})$ [11].

Now suppose that $m(G) > 0$ and we have shown (*) for all grammars G' such that $m(G') < m(G)$. Call a symbol X in G *quasi-linear* if whenever $X \Rightarrow^{L^*} uYv$ and $Y \in V - \Sigma - I$, then u is in $(\Sigma \cup I)^*$ and v is in Σ^*. Let Q be the set of quasi-linear nonterminals. We shall show

(**) there is a grammar G_1 with $m(G_1) < m(G)$, a finite nonempty substitution τ and homomorphisms h_1 and h_2 and a new symbol c such that for any control set C, $L(G_1, C) = f_{\infty,c}(h_1, h_2, L(G_1, \tau(C))c)$.

The desired result, (*), then follows from (**) and the induction hypothesis.

Notice that if $S \Rightarrow^{L*} uYv$, u in $(\Sigma \cup I)^*$, $Y \in V - \Sigma - I$, and Yv has $m(G)$ nonterminals, then Y must be quasi-linear. If every symbol quasi-linear in G becomes finite state in G_1, then $m(G_1) \leqslant m(G) - 1$.

We can divide the rule names of G into Δ and Δ' where Δ contains X-rules for X in Q and Δ' contains X-rules for X not in Q. Notice that any symbols reachable from quasi-linear symbols are either quasi-linear or terminal. So if X is in Q and $X \Rightarrow^{\pi} w$, then π is in Δ^*.

Our construction blends two different ideas. Call a symbol X in Q *linear* if whenever $X \overset{*}{\Rightarrow} u$ then u is in $\Sigma^*(V \cup \{e\})\,\Sigma^*$. The construction in Theorem 6.1 turns on defining homomorphisms h_1 and h_2 such that whenever X is linear and $X \Rightarrow^{\pi} w$ for a terminal string w, then $w = h_1(\pi)\, h_2(\pi^R)$. We shall modify this construction to allow X to be quasi-linear instead of just linear. In this process we have to add new nonterminals and productions to account for the fact that our intermediate derivation strings might contain more than one nonterminal. This will be the role of the substitution τ. If $X \Rightarrow^{\pi} w$ in G for X quasi-linear and w a terminal string, then there will be $\hat{\pi}$ in $\tau(\pi)$ such that $w = h_1(\hat{\pi})\, h_2(\hat{\pi}^R)$. Using $f_{\infty,c}$ allows one to play this game as often as needed, to handle any number of control subwords from Δ^+.

This does not work for nonterminals not in Q. We shall let G_1 simulate directly the rules of Δ' with the same names, so $\tau(p) = \{p\}$ for p in Δ'. Whenever a control subword π in Δ^+ is encountered, G_1 will use some control word $\hat{\pi}$ in $\tau(\pi)$ (and $\hat{\pi}$ will in fact be unique) and finite state symbols to generate $\hat{\pi}$ itself (flanked by c's) and $h_1(\hat{\pi})\, h_2(\hat{\pi}^R)$ will be the terminal string generated in G by the subderivation controlled by π. Symbols quasi-linear in G become finite state in G_1 so we will have $m(G_1) \leqslant m(G) - 1$.

Thus a derivation in G of the form

$$\begin{aligned} S &\overset{\lambda_1}{\Rightarrow} w_1X_1u_1 \overset{\pi_1}{\Rightarrow} w_1z_1u_1 \overset{\lambda_2}{\Rightarrow} w_1z_1w_2X_2u_2 \Rightarrow \cdots \overset{\lambda_r}{\Rightarrow} w_1z_1w_2z_2 \cdots w_rX_ru_r \\ &\overset{\pi_r}{\Rightarrow} w_1z_1w_2z_2 \cdots w_rz_ru_r \overset{\lambda_{r+1}}{\Rightarrow} w_1z_1w_2z_2 \cdots w_rz_rw_{r+1} \end{aligned}$$

with each λ_i in $(\Delta')^*$, X_i quasi-linear, π_i in Δ^*, and w_i and z_i terminal strings, will correspond in G_1 to the derivation

$$S \overset{\lambda_1\hat{\pi}_1\lambda_2\hat{\pi}_2\cdots\lambda_r\hat{\pi}_r\lambda_{r+1}}{\Longrightarrow} w$$

for $w = w_1c\hat{\pi}_1cw_2c\hat{\pi}_2c \cdots w_rc\hat{\pi}_rcw_{r+1}$ with each $\hat{\pi}_i$ in $\tau(\pi_i)$. Homomorphism h_1 will be defined to be the identity on Σ and h_2 will erase all of Σ. Thus

$$\begin{aligned} f_{\infty,c}(h_1, h_2, \{wc\}) &= \{h_1(w_1)\, h_2(w_1^R)\, h_1(\hat{\pi}_1)\, h_2(\hat{\pi}_1^R) \cdots \\ &\qquad h_1(w_r)\, h_2(w_r^R)\, h_1(\hat{\pi}_r)\, h_2(\hat{\pi}_r^R)\, h_1(w_{r+1})\, h_2(w_{r+1}^R)\} \\ &= \{w_1h_1(\hat{\pi}_1)\, h_2(\hat{\pi}_1^R) \cdots w_rh_1(\hat{\pi}_r)\, h_2(\hat{\pi}_r^R)w_{r+1}\} \\ &= \{w_1z_1 \cdots w_rz_rw_{r+1}\}. \end{aligned}$$

If $\lambda_1\pi_1 \cdots \lambda_r\pi_r\lambda_{r+1}$ is in C then $\lambda_1\hat{\pi}_1 \cdots \lambda_r\hat{\pi}_r\lambda_{r+1}$ will be in $\tau(C)$. Hence we will have

$$L(G, C) \subseteq f_{\infty,c}(h_1, h_2, L(G_1, \tau(C))c).$$

On the other hand we will set up G_1 to generate only strings of this form and hence get $L(G, C) = f_{\infty,c}(h_1, h_2, L(G_1, \tau(C))c)$ to establish (**).

It remains only to give the construction of G_1. We shall define h_1, h_2, and the production set P_1 of G_1 simultaneously. Let $t = \text{Max}(\{1\} \cup \{|y| \mid \exists Z \to y \text{ in } P\})$. For each word v in $(\Sigma \cup I)^*Q$ with $1 \leqslant |v| \leqslant t$, let $[v]$ be a new nonterminal and let V_1 be the set of all these symbols. Let c be a new symbol. The terminal set of G_1 will be $\Sigma \cup \tau(\Delta) \cup \{c\}$ and the nonterminal set will be $V_1 \cup (V - \Sigma)$. We shall have $h_1(a) = a$ and $h_2(a) = e$ for a in Σ.

For a rule

$$p\colon X \to u$$

in P with u in $(V - Q)^*$ and X not in Q, let P_1 contain this rule with the same name and let $\tau(p) = \{p\}$.

For a rule

$$p\colon X \to u_1 Y_1 \cdots u_r Y_r u_{r+1}$$

in P with X not in Q, u_i in $(V - Q)^*$, $1 \leqslant i \leqslant r + 1$ and Y_i in Q, $1 \leqslant i \leqslant r$, $r \geqslant 1$, let P_1 contain with the same name, p, the rule

$$p\colon X \to u_1 c Y_1 \cdots u_r c Y_r u_{r+1}$$

and let $\tau(p) = \{p\}$.

For a rule

$$p\colon X \to \alpha u \beta$$

with X in Q, u in $(\Sigma \cup Q)^+ - \Sigma V^* - V^*\Sigma$, and α, β in Σ^*, let P_1 contain the rule

$$p\colon X \to p[u]$$

plus for each v such that $[Xv]$ and $[uv]$ are legitimate, the rule

$$p(v)\colon [Xv] \to p(v)[uv]$$

with $\tau(p) = \{p\} \cup \{p(v) \mid [Xv] \text{ and } [uv] \text{ are legitimate}\}$ and $h_1(p') = \alpha$ and $h_2(p') = \beta$ for each p' in $\tau(p)$.

For a rule of G

$$p\colon X \to \alpha$$

with X in Q and α in Σ^* let P_1 contain a rule

$$p\colon X \to pc$$

and a rule

$$p(e)\colon [X] \to p(e)c$$

plus for each βv such that $[X\beta v]$ is a legitimate symbol, β is in Σ^* and v is in $(\Sigma \cup Q)^+ - \Sigma V^*$, the rule

$$p(\beta v)\colon [X\beta v] \to p(\beta v)[v]$$

with $\tau(p) = \{p\} \cup \{p(v) \mid [Xv] \text{ is legitimate}\}$ and $h_1(p) = \alpha$ and $h_1(p(\beta v)) = \alpha\beta$ and $h_2(p') = e$ for all p' in $\tau(p)$.

We leave it to the reader to verify that G_1, h_1, h_2, and τ so defined have the required properties. ∎

COROLLARY 7.11. *For any* semiAFL $\mathscr{L}$ *and any* $n \geqslant 2$,

$$\text{CONTROL}_\infty(\text{LDBnG}, \mathscr{L}) = \text{IHOM}_\infty(\mathscr{L}).$$

Notice that if $\mathscr{L}$ is a full semiAFL contained in the family of context-sensitive languages then by Theorem 2.11 and Corollary 7.11, so is $\text{IHOM}_\infty(\mathscr{L})$. Thus by Theorem 4.4, if $\text{CONTROL}(\mathscr{G}(G), \mathscr{L}) \subseteq \text{IHOM}_\infty(\mathscr{L})$ for every full semiAFL $\mathscr{L}$, then G must be c-equivalent to a left derivation bounded grammar.

COROLLARY 7.12. *A grammar G is c-equivalent to a left derivation bounded grammar if and only if* $\text{CONTROL}(\mathscr{G}(G), \mathscr{L}) \subseteq \text{IHOM}_\infty(\mathscr{L})$ *for every full* semiAFL $\mathscr{L}$.

We can give a partial analog of Corollary 6.7 for $\text{IHOM}(\mathscr{L})$, based on the notion of iterative languages, languages for which a particular type of intercalation theorem holds. Let us give the necessary definitions.

DEFINITION. Let $k \geqslant 1$. A language L is k-iterative if there is an integer $k_1 \geqslant 1$ such that for all w in L with $|w| > k_1$, there are $u_1, \ldots, u_{k+1}, v_1, \ldots, v_k$ with $w = u_1v_1 \cdots u_kv_ku_{k+1}$, $v_1v_2 \cdots v_k \neq e$, and $u_1v_1^n \cdots u_kv_k^nu_{k+1}$ in L for all $n \geqslant 0$. A language L is *weakly k-iterative* if either L is finite or it contains a k-iterative subset. A family of languages $\mathscr{L}$ is *k-iterative* (*weakly k-iterative*) if every member of $\mathscr{L}$ is *k-iterative* (weakly k-iterative).

DEFINITION. For a grammar $G = (V, \Sigma, P, S)$ the *left derivation language* of G is

$$\text{LD}(G) = \{x \mid \text{for some } w \text{ in } \Sigma^*,\ S \overset{x}{\Rightarrow} w\}.$$

The key lemma we need is the following.

LEMMA 7.13. *Let G be a monotonic reduced left derivation bounded grammar. Suppose that for $s \geqslant 1$,* $\text{LD}(G)$ *contains a subset*

$$C = \{u_1v_1^n \cdots u_sv_s^nu_{s+1} \mid n \geqslant 0\}$$

where $v_i \neq e$, $1 \leqslant i \leqslant s$. Then $L(G, C)$ is weakly $2s$-iterative.

Proof. For each n, let $z(n) = u_1v_1^n \cdots u_sv_s^nu_{s+1}$, $z(n, i) = u_1v_1^n \cdots u_i$, and $\bar{z}(n, i) = u_1v_1^n \cdots u_iv_i^n$ for $1 \leqslant i \leqslant s$. Let $S \Rightarrow^{z(n)} w(n)$, $S \Rightarrow^{z(n,i)} w(n, i)$, and $S \Rightarrow^{\bar{z}(n,i)} \bar{w}(n, i)$. Thus $L(G, C) = \{w(n) \mid n \geqslant 0\}$. We wish to show that $L = \{w(n) \mid n \geqslant 2\}$ is $2s$-iterative. We can regard the v_i as s-iterative factors of C. In the same sense, we wish to pick out $2s$ iterative factors for L. We do so by considering the possibilities for $w(n, i)$ and $\bar{w}(n, i)$ for each i. Let $G = (G, \Sigma, P, S)$.

Since C is a subset of LD(G), each $z(n)$ actually controls a complete left-to-right derivation of a member of $L(G)$. Each v_i is nonempty and starts with a rule with left-hand side say Z_i. After applying $z(n, i)$, it must be possible to apply n copies of v_i, for each n. Since G is left derivation bounded, the Z_i's cannot pile up during $z(n, i)$ but must be generated during v_i. Recall that if $X \Rightarrow^z xXy$ in G, then y must be in Σ^*; if we can immediately repeat z then we must have xy in Σ^+ (G is monotonic). These considerations tell us that there are only two possibilities for $w(n, i)$ and v_i, depending on whether v_i applies only to Z_i and descendants or to Z_i plus other symbols already present in $w(n, i)$, but to be generated anew during v_i. Here are the cases.

(1) We have $Z_i \Rightarrow^{v_i} x_i Z_i y_i$. Then $x_i y_i$ is in Σ^+ and for all $n \geqslant 0$, $w(n, i) = w(n, i, 1)\, Z_i w(n, i, 2)$. Thus $\bar{w}(n, i) = w(n, i, 1)\, x_i^n Z_i\, y_i^n w(n, i, 2)$ for all $n \geqslant 0$, so x_i and y_i certainly are two iterative factors for L, and at least one is nonempty.

(2) Z_i generates a terminal string under some proper initial substring of v_i. Thus some further string of nonterminals in $w(n, i)$ is needed to generate Z_i and itself. There must be $Z_{i1} = Z_i, Z_{i2}, \ldots, Z_{ip}$ for $p \geqslant 2$ and $v_i = v_{i1} \cdots v_{ip}$, each $v_{ij} \neq e$ such that for all $n \geqslant 0$, $w(n, i) = w(n, i, 1)\, Z_{i1} \cdots w(n, i, p)\, Z_{ip} w(n, i, p + 1)$ and $w(n, i, j) \in \Sigma^*$ for $1 \leqslant j \leqslant p$, and $Z_{ij} \Rightarrow^{v_{ij}} x_{ij}$ in Σ^+ for $1 \leqslant j \leqslant p - 1$, and $Z_{ip} \Rightarrow^{v_{ip}} y_{i1} Z_{i1} \cdots y_{ip} Z_{ip} y_{i(p+1)}$ each y_{ij} in Σ^*. Let $x_i = y_{i1} x_{i1} \cdots y_{i(p-1)} x_{i(p-1)}\, y_{ip}$ and $y_i = y_{i(p+1)}$.

Clearly $x_i \neq e$. Now observe that

$$\bar{w}(0, i) = w(0, i) = w(0, i, 1)\, Z_{i1} \cdots w(0, i, p)\, Z_{ip} w(0, i, p + 1),$$

$$\bar{w}(1, i) = w(1, i, 1)\, z_{i1} \cdots w(1, i, p - 1)\, x_{i(p-1)} w(1, i, p)\, y_{i1} Z_{i1} \cdots Z_{ip}\, y_{i(p+1)} w(1, i, p + 1),$$

and for $n \geqslant 1$

$$\bar{w}(n + 1, i) = w(n + 1, i, 1) \cdots w(n + 1, i, p) x_i^n y_{i1} Z_{i1} \cdots Z_{ip}\, y_{i(p+1)}\, y_i^n w(n + 1, i, p + 1).$$

Hence x_i and y_i will be iterative factors for L.

Thus in all cases each v_i yields two iterative factors for L at least one of which is nonempty. These factors $x_1, y_1, \ldots, x_s, y_s$ can interweave in $w(2)$, subject to the conditions that x_i always precedes y_i and if x_{i+1} precedes y_i so does y_{i+1}. ∎

The following lemma is stated without proof; we have implicitly used it before in manipulating left derivation bounded grammars.

LEMMA 7.14. *Let G be a reduced context-free grammar. Then* LD(G) *is regular if and only if G is left derivation bounded.*

This gives us the iteration theorem and its corollary, the weak hierarchy theorem.

THEOREM 7.15. *Let $\mathscr{L}$ be a weakly k-iterative full* semiAFL. *Then* CONTROL (LDBG, $\mathscr{L}$) *is weakly $2k$-iterative.*

Proof. By Theorem 2.9, we need only consider languages $L(G, C)$ for G a reduced monotonic left derivation bounded grammar and C in $\mathscr{L}$. By Lemma 7.14, LD(G) is

regular and so $C_1 = \mathrm{LD}(G) \cap C$ is in $\mathscr{L}$. Clearly $L(G, C) = L(G, C_1)$. If C_1 is finite, then $L(G, C)$ is finite and a fortiori weakly $2k$-iterative. Otherwise, since $\mathscr{L}$ is weakly k-iterative, C_1 contains a subset $C_2 = \{u_i v_1^n \cdots u_s v_s^n u_{s+1} \mid n \geqslant 0\}$ with $1 \leqslant s \leqslant k$ and each $v_i \neq e$. By Lemma 7.13, $L(G, C_1)$ is weakly $2s$-iterative and hence so is $L(G, C_1) = L(G, C)$. Thus $L(G, C)$ is weakly $2k$-iterative. ∎

THEOREM 7.16. *Let $\mathscr{L}$ be a weakly k-iterative full* semiAFL. *Let* $L_k = \{a_1^n \cdots a_k^n \mid n \geqslant 1\}$.

(1) *For all* $r \geqslant 1$, $\mathrm{CONTROL}_r(\mathrm{LDBG}, \mathscr{L}) \subsetneq \mathrm{CONTROL}_{r+1}(\mathrm{LDBG}, \mathscr{L})$ *and* $\mathrm{IHOM}_r(\mathscr{L}) \subsetneq \mathrm{IHOM}_{r+1}(\mathscr{L})$.

(2) *If* L_k *is in* $\mathscr{L}$, *then for all* $r \geqslant 1$,

$$\mathrm{CONTROL}_{r+1}(\mathrm{LINEARG}, \mathscr{L}) - \mathrm{CONTROL}_r(\mathrm{LDBG}, \mathscr{L}) \neq \phi.$$

Proof. In view of Lemma 7.9 and Theorem 7.10, to show (1), it suffices to show that for each $r \geqslant 1$ there is an $r' \geqslant r$ with $\mathrm{CONTROL}_{r'}(\mathrm{LDBG}, \mathscr{L}) \neq \mathrm{CONTROL}_r(\mathrm{LDBG}, \mathscr{L})$. Since $\mathscr{L}$ is weakly k-iterative, by Theorem 7.15, $\mathrm{CONTROL}_r(\mathrm{LDBG}, \mathscr{L})$ must be weakly $2^r k$-iterative. The language $L_{2^r k+1}$ is not weakly $2^r k$-iterative and so is not in $\mathrm{CONTROL}_r(\mathrm{LDBG}, \mathscr{L})$, although it clearly is in $\mathrm{CONTROL}_{rk+1}(\mathrm{LINEARG}, \mathrm{REGULARL})$, and so in $\mathrm{CONTROL}_{rk+1}(\mathrm{LINEARG}, \mathscr{L})$. To show (2), one need only observe that $L_{2^r k+1}$ is in $\mathrm{CONTROL}_{r+1}(\mathrm{LINEARG}, \hat{\mathscr{M}}(L_k))$.

COROLLARY 7.17. *Let $\mathscr{L}$ be a full* semiAFL *contained in* CFL.

(1) *For each* $r \geqslant 1$, $\mathrm{CONTROL}_r(\mathrm{LDBG}, \mathscr{L}) \subsetneq \mathrm{CONTROL}_{r+1}(\mathrm{LDBG}, \mathscr{L})$ *and* $\mathrm{IHOM}_r(\mathscr{L}) \subsetneq \mathrm{IHOM}_{r+1}(\mathscr{L})$.

(2) *If* L_2 *is in* $\mathscr{L}$, *then for each* $r \geqslant 1$,

$$\mathrm{CONTROL}_{r+1}(\mathrm{LINEARG}, \mathscr{L}) - \mathrm{CONTROL}_r(\mathrm{LDBG}, \mathscr{L}) \neq \phi.$$

COROLLARY 7.18. *For each* $r \geqslant 1$,

$$\mathrm{CONTROL}_r(\mathrm{LDBG}, \mathrm{REGULARL}) \subsetneq \mathrm{CONTROL}_r(\mathrm{LDBG}, \mathrm{REGULARL}),$$
$$\mathrm{IHOM}_r(\mathrm{REGULARL}) \subsetneq \mathrm{IHOM}_{r+1}(\mathrm{REGULARL}),$$
$$\mathrm{CONTROL}_{r+1}(\mathrm{LINEARG}, \mathrm{REGULARL}) - \mathrm{CONTROL}_r(\mathrm{LDBG}, \mathrm{REGULARL}) \neq \phi,$$

and $\mathrm{CONTROL}_{r+1}(\mathrm{LINEARG}, \mathrm{REGULARL})$ *and* $\mathrm{CONTROL}_r(\mathrm{LDBG}, \mathrm{REGULARL})$ *are incomparable. Further,* $\mathrm{IHOM}(\mathrm{REGULARL})$ *and* $\mathrm{CONTROL}(\mathrm{LINEARG}, \mathrm{REGULARL})$ *are incomparable.*

Corollary 7.18 also holds with CFL substituted for REGULARL everywhere.

8. CONCLUSIONS AND OPEN PROBLEMS

We have seen that the nature of the G-control operator depends critically on whether or not G is left derivation bounded. If G is inherently non-ldb (i.e., is not c-equivalent to any ldb grammar), then one application of the G-control operator takes one from the

linear context-free languages to the recursively enumerable languages. If G is left derivation bounded such a jump in complexity cannot occur; if $\mathscr{L}$ is a full semiAFL contained in, say, the family of context-sensitive languages, then the closure of $\mathscr{L}$ under the G-control operator is also contained in the family of context-sensitive languages.

If G is c-equivalent to a grammar in Greibach Normal Form then for every full semiAFL $\mathscr{L}$, CONTROL$(\mathscr{G}(G), \mathscr{L}) = \mathscr{H}(\mathscr{L}(G) \wedge \mathscr{L})$, and this is a necessary and sufficient condition for such a characterization of CONTROL$(\mathscr{G}(G), \mathscr{L})$ to hold. This characterization is restricted to grammars that are not left derivation bounded and cannot hold for a left derivation bounded grammar which is inherently nonregular (not c-equivalent to any regular grammar). For left derivation bounded grammars we obtained a different type of characterization of CONTROL$(\mathscr{G}(G), \mathscr{L})$ in terms of homomorphic replication. We saw that CONTROL$_\infty(\mathscr{G}(G), \mathscr{L}) = \text{HOM}_\infty(\mathscr{L})$, the closure of $\mathscr{L}$ under homomorphic replication, for every full semiAFL $\mathscr{L}$ if and only if G is c-equivalent to a nonterminal bounded self-embedding grammar. For G self-embedding and nonterminal bounded, the G-control operator is hierarchical in the sense that if $\mathscr{L} \neq \text{CONTROL}(\mathscr{G}(G), \mathscr{L})$ for any full semiAFL $\mathscr{L}$, then CONTROL$_k(\mathscr{G}(G), \mathscr{L}) \neq$ CONTROL$_{k+1}(\mathscr{G}(G), \mathscr{L})$ for all $k \geqslant 0$. For the general left derivation bounded case we obtained a characterization of CONTROL$(\mathscr{G}(G), \mathscr{L})$ in terms of infinite homomorphic replications.

Several questions remain open. We saw that if G is left derivation bounded and nontrivial then CONTROL$(\mathscr{G}(G), \mathscr{L})$ is full semiAFL whenever $\mathscr{L}$ is a semiAFL (and indeed only closure of $\mathscr{L}$ under e-free regular substitution and under union was needed). It is an open question whether this holds for every nontrivial grammar G; if any counter-example exists, G must be inherently non-ldb, and not c-equivalent to any grammar in Greibach Normal Form, and $\mathscr{L}$ cannot lie between LINEARL and RE. It is also open whether we have CONTROL $(\mathscr{G}(G), \mathscr{L}) = \mathscr{H}(\mathscr{L}(G) \wedge \mathscr{L})$ for every full semiAFL $\mathscr{L}$ whenever G is self-embedding and inherently non-ldb.

Another research problem is to consider control sets on arbitrary derivations, not just left-to-right derivations. Some of these results can be carried over without much difficulty while others (e.g. the characterizations of CONTROL$(\mathscr{G}(G), \mathscr{L})$ in Sections 5–7) would appear to have no reasonable analog.

One question that has not been considered in this paper is whether c-equivalence is decidable for context-free grammars. There is reason to believe that g-equivalence is decidable for context-free grammars.[8] Thus the question to investigate is:

(*) Given two g-equivalent grammars G and H, are G and H c-equivalent?

There are four separate cases to consider for G (and H). (1) G is c-equivalent to a nonterminal bounded grammar; (2) G is c-equivalent to a left derivation bounded grammar but not to any nonterminal bounded grammar; (3) G is not c-equivalent to any left derivation bounded grammar but $\mathscr{L}(G) \neq$ CFL; (4) $\mathscr{L}(G) =$ CFL. The four cases are mutually exclusive. The results in this paper show that it is decidable which case occurs since the various necessary conditions on a nonterminal are known to be decidable: self-

[8] J. Goldstine, private communication.

embedding, right expansive, pse, nontrivial, finite state, linear, quasi-linear, left derivation bounded, nonterminal bounded, etc. All grammars falling under case (4) are c-equivalent. We conjecture that cases (1) and (2) can be decided by a detailed analysis of the existence and relationship of self-embedding, left self-embedding and right self-embedding symbols. Case (3) is more complex; it might be that here too g-equivalence implies c-equivalence among grammars satisfying case (3).

Another question which has not been considered here is when the G-control operator takes full principal semiAFLs into full principal semiAFLs. We know $\mathscr{L}(G)$ is always full principal for G nontrivial [5]. If CONTROL$(\mathscr{G}(G), \mathscr{L}) = \mathscr{K}(\mathscr{L}(G) \wedge \mathscr{L})$ for every full semiAFL $\mathscr{L}$ (which by Theorem 5.14 means that G is c-equivalent to a grammar in Greibach Normal Form), then the G-control operator takes full principal semiAFLs into full principal semiAFLs [8, 31]. The G-control operator also has this property when G is metalinear as can be seen by careful study of the characterization in Theorem 6.1. The results of Sections 6 and 7 strongly suggest that this property can be shown to hold for any left derivation bounded grammar by a careful analysis of cases. Indeed, one is led to conjecture that CONTROL$(\mathscr{G}(G), \mathscr{L})$ is a full principal semiAFL whenever $\mathscr{L}$ is a full principal semiAFL. The difficulty in proving such a conjecture or providing a counterexample is indicated by the facts that (1) we do not yet even have a proof that CONTROL$(\mathscr{G}(G), \mathscr{L})$ is always a full semiAFL and (2) for G inherently non left derivation bounded and LINEARL $\subseteq \mathscr{L} \subseteq$ RE, CONTROL$(\mathscr{G}(G), \mathscr{L}) =$ RE which is full principal, but the proof of Theorem 4.4 certainly gives no clue to finding a generator for RE from a generator for $\mathscr{L}$!

Proposition 2.13 suggests the following question. When can a context-free full principal semiAFL be characterized as CONTROL$(\mathscr{G}_S(G)$, REGULARL)? Cremers and Ginsburg [5, 32] have considered when a context-free full principal semiAFL can be characterized as $\mathscr{L}(G)$, which by Proposition 2.2 equals CONTROL$(\mathscr{G}(G)$, REGULARL).

It is not known how far Theorem 7.16 can be extended. In particular, is there a full semiAFL $\mathscr{L}$ with $\mathscr{L} \neq$ IHOM$(\mathscr{L})$ but IHOM$_\infty(\mathscr{L}) =$ IHOM$_k(\mathscr{L})$ for some $k \geqslant 1$?

A final question, suggested by Theorems 2.11, 6.1, and 7.10. concerns full semiAFLs contained in $\mathscr{P}$, the family of languages accepted in polynomial time by deterministic multitape Turing machines. If $\mathscr{L}$ is a full semiAFL contained in $\mathscr{P}$, is HOM$(\mathscr{L})$ also contained in $\mathscr{P}$? What about IHOM$(\mathscr{L})$? It is known that IHOM$_\infty$(CFL) is contained in $\mathscr{P}$ [33].

References

1. B. S. Baker and R. V. Book, Reversal-bounded multipushdown machines, *J. Comput. System Sci.* **8** (1974), 315–332.
2. R. B. Banerji, Phrase structure languages, finite machines and channel capacity, *Inform. Contr.* **6** (1963), 153–162.
3. R. V. Book and S. A. Greibach, Quasi-realtime languges, *Math. Systems Theory* **4** (1970), 97–111.
4. N. Chomsky, On certain formal properties of grammars, *Inform. Contr.* **2** (1959), 137–167.
5. A. Cremers and S. Ginsburg, Context-free grammar forms, *J. Comput. System Sci.* **11** (1975), 86–117.

6. A. GABRIELIAN AND S. GINSBURG, Grammar schemata, *J. Assoc. Comput. Mach.* **21** (1974), 213–226.
7. S. GINSBURG AND S. A. GREIBACH, Abstract Families of Languages, *in* "Memoirs of the American Mathematical Society," No. 87, pp. 1–32, 1969.
8. S. GINSBURG AND S. A. GREIBACH, Principal AFL, *J. Comput. System Sci.* **4** (1970), 308–338.
9. S. GINSBURG, S. A. GREIBACH, AND M. A. HARRISON, One-way stack automata, *J. Assoc. Comput. Mach.* **14** (1967), 389–418.
10. S. GINSBURG AND E. H. SPANIER, AFL with the semilinear property, *J. Comput. System Sci.* **5** (1971), 365–396.
11. S. GINSBURG AND E. H. SPANIER, Control sets on grammars, *Math. Systems Theory* **2** (1968), 159–178.
12. S. GINSBURG AND E. H. SPANIER, Derivation-bounded languages, *J. Comput. System Sci.* **2** (1968), 228–250.
13. S. GINSBURG AND E. H. SPANIER, Finite-turn pushdown automata, *SIAM J. Control* **4** (1966), 429–453.
14. S. A. GREIBACH, Checking automata and one-way stack languages, *J. Comput. System Sci.* **3** (1969), 196–217.
15. S. A. GREIBACH, An infinite hierarchy of context-free languages, *J. Assoc. Comput. Mach.* **16** (1969), 91–106.
16. S. A. GREIBACH, A new normal-form theorem for context-free phrase structure grammars, *J. Assoc. Comput. Mach.* **12** (1965), 42–52.
17. S. A. GREIBACH, Syntactic operators on full semiAFLS, *J. Comput. System Sci.* **6** (1972), 30–76.
18. S. A. GREIBACH, The unsolvability of the recognition of linear context-free languages, *J. Assoc. Comput. Mach.* **13** (1966), 582–587.
19. S. A. GREIBACH AND J. HOPCROFT, Independence of AFL operators, *in* "Studies in Abstract Families of Languages" (Ginsburg, Greibach, and Hopcroft, Eds.), *Memoirs of the American Mathematical Socity*, No. 87, pp. 33–40, 1969.
20. S. A. GREIBACH AND J. HOPCROFT, Scattered context grammars, *J. Comput. System Sci.* **3** (1969), 233–247.
21. N. A. KHABBAZ, Control sets on linear grammars, *Inform. Contr.* **25** (1974), 206–221.
22. N. A. KHABBAZ, A geometric hierarchy of languages, *J. Comput. System Sci.* **8** (1974), 142–157.
23. D. J. ROSENKRANTZ, Programmed grammars and classes of formal languages, *J. Assoc. Comput. Mach.* **16** (1969), 107–131.
24. A. SALOMAA, "Formal Languages," Academic Press, New York, 1973.
25. A. SALOMAA, On grammars with restricted use of productions, *Ann. Acad. Sci. Fenn. Ser. A I* **454** (1969).
26. A. SALOMAA, Periodically time-variant context-free grammars, *Inform. Contr.* **17** (1970), 294–311.
27. R. SIROMONEY, Finite-turn checking automata, *J. Comput. System Sci.* **5** (1971), 549–559.
28. S. J. WALLJASPER, Left-derivation bounded languages, *J. Comput. System Sci.* **8** (1974), 1–7.
29. E.-M. WOTSCHKE, "Ordered Grammars with Equivalence Classes: Some Formal and Linguistic Aspects," Ph.D. Dissertation, University of California at Los Angeles, 1975.
30. S. GINSBURG AND S. A. GREIBACH, Mappings which preserve context-sensitive languages, *Inform. Contr.* **9** (1966), 563–582.
31. S. GINSBURG AND S. A. GREIBACH, Multitape AFA, *J. Assoc. Comput. Mach.* **19** (1972), 193–221.
32. A. CREMERS AND S. GINSBURG, Characterizations of context-free grammatical families, *in* "Proceedings 15th Annual IEEE Symposium on Switching and Automata Theory," pp. 199–204, October 1974, New Orleans.
33. S. A. GREIBACH, One-way finite visit automata, *Theoretical Computer Science*, to appear.